GRAFISCHE EXPERIMENTEN VOOR MSX-COMPUTERS

MET BEELDBEWERKINGSPROJECT

W.H.M. van Dreumel



Kluwer

W.H.M. van Dreumel GRAFISCHE EXPERIMENTEN VOOR MSX-COMPUTERS

Grafische experimenten voor MSX-computers

met beeldbewerkingsproject

W. H. M. van Dreumel

Grafische experimenten voor MSX-computers

met beeldbewerkingsproject

Kluwer Technische Boeken B.V.
Deventer-Antwerpen

Omslag: W. Niessink.
Illustraties: A. J. van Weij

ISBN 90 201 1967 2
D/1987/0108/159
NUGI 434

1e druk 1987

Woord vooraf

MSX, Microsoft Extended BASIC. Een programmeertaal die de homecomputer mogelijkheden geeft op hoog niveau. Meer dan honderdvijftig sleutelwoorden brengen de computer op een niveau waarvan de thuiscomputeraar(ster) tot voor kort nog slechts kon dromen.
Maar MSX is meer dan een uitgebreide programmeertaal. Het is een standaard voor computerhardware en -software. Aangepast aan de behoefte van een groot publiek. Het comfortabele toetsenbord laat langdurig intensief gebruik toe. Een handige cursorbesturing maakt het bedienen tot een plezier. Maar de MSX-computer is ook volledig uitgerust om de meer passieve gebruik(st)er van dienst te zijn. Spelcassettes kunnen er zo worden ingestoken. Joystick-ingangen completeren het spelplezier. Kant en klare software zal in overweldigende hoeveelheden op ons afkomen.
Onder de programmeurs en de spelletjesgeniet(st)ers bevindt zich een grote groep die de uitdaging van de computer aanvaard hebben. Worstelend banen zij zich een weg door boeken en hobbytijdschriften. Op zoek naar de oplossing van probleempjes waarvoor de handleiding geen oplossing geeft. Aardige kleine programma's worden rechtstreeks uit tijdschriften overgenomen. Zelf verzonnen oplossingen worden geprobeerd en ergens op een bandje gezet. Na verloop van tijd vinden we in de buurt van iedere computer wel een stapeltje cassettes vol met van die werkstukjes die te leuk zijn om zo maar weg te gooien. Ze zijn vaak al wat stoffig omdat ze niet zo erg veel worden gebruikt. Toch is er in veel gevallen heel wat tijd in die bandjes gestoken. Plezierig doorgebracht weliswaar, maar toch jammer om dat zomaar ongebruikt te laten liggen.
In het eerste deel van het boek poetsen we het stof van mijn grafische experimentele programmaatjes. Alle probeerseltjes worden als pareltjes opgewreven en aan een snoer geregen, tegelijkertijd leren we de vele grafische mogelijkheden van de MSX-computer kennen. De cassetteband wordt als rijgsnoer gebruikt. Een rijgsnoer, lang genoeg om een flinke keten te vormen.
In het tweede deel van dit boek gaan we wat serieuzer aan het werk. We bouwen een beeldopnemer waarmee we beelden kunnen digitaliseren. Te zamen met de nodige software ontstaat er een prachtig beeldverwerkingssysteem.

W.H.M. van Dreumel

Inhoud

Deel 1
Grafische toepassingen

1 Enkele opmerkingen vooraf

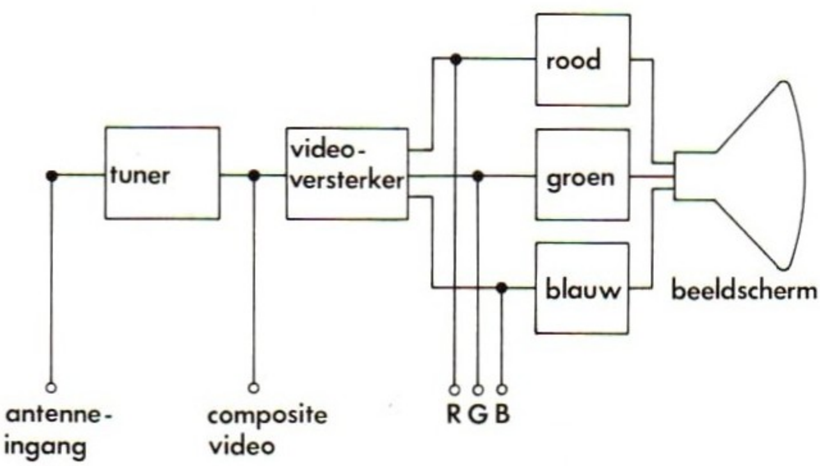
Om te beginnen enkele opmerkingen. Dit boek is geen programmeercursus. Toch worden er heel wat lesjes in programmeren gegeven. Het lezen van dit boek kan dan ook het beste achter de computer gebeuren. Door de programma's direct in te typen en het effect op het scherm te bekijken, zal het grafisch gebruik van de MSX-computer voor u, na het doorwerken van dit deel van het boek, geen geheimen meer hebben. Om het gebruik van de meer ingewikkelde opdrachten eenvoudiger te maken, is een deel ervan apart in hoofdstuk 17 opgenomen. Ze zijn hierbij naar functie gerangschikt. Dus niet alfabetisch zoals in de meeste handleidingen en leerboeken het geval is. Het opzoeken wordt hierdoor vereenvoudigd.

Listings

De manier waarop de programmalistings weergegeven zijn, is enigszins afwijkend. Spaties die normaal gesproken mogen worden weggelaten, zijn in dit boek wél weergegeven. Dit bevordert de overzichtelijkheid van de programma's aanzienlijk zodat minder gemakkelijk typefouten zullen worden gemaakt. Het is niet belangrijk of de afgebeelde spaties wel of niet worden overgenomen. Uw machine regelt zelf wel wat hij ervan gebruikt. Het programma op het scherm kan er dus iets anders uitzien dan de programmalistings in dit boek.

Beeldscherm

Gebruik bij voorkeur een kleurenmonitor of een kleuren-TV met een aparte



Afbeelding 1-1 De aansluitmogelijkheden van een kleurenmonitor.

video-ingang. Op moderne TV's is soms een RGB-plug aanwezig. De MSX-computer heeft hiervoor echter geen standaardaansluiting. Afbeelding 1-1 laat het verschil zien tussen de diverse aansluitmogelijkheden. Hoe minder TV-onderdelen het computersignaal hoeft te passeren, hoe beter de beeldkwaliteit zal zijn.
[Afbeelding 1-1]

Cassetterecorder

Het opslaan van programma's op cassette gaat bij de MSX-computer bijzonder handig, doordat de recordermotor vanuit de computer kan worden gestuurd. Er is niets op tegen een gewone audiocassetterecorder voor de opslag van programma's te gebruiken. Voordat de MSX-computer gegevens naar de recorder stuurt, geeft hij een inregelsignaal van ongeveer vijf seconden. Gedurende deze tijd kan de automatische opnamesterkteregeling, die in de meeste audiorecorders aanwezig is, zichzelf instellen. De weergavesterkte moet met de hand worden ingesteld. Enig proberen zal in de meeste gevallen een bevredigende stand van de volumeregelaar opleveren. De volumeregeling dient vrij ver opgedraaid te zijn.
Ideaal is het gebruik van een data-recorder. In wezen is er geen verschil met een gewone audiorecorder. De opname- en weergavesterkte zijn in de stand DATA vast ingesteld. In veel gevallen kan de fase van het signaal worden omgedraaid. Enig proberen geeft ook hier snel resultaat. Bij het uitwisselen van bandjes kan het gebeuren dat programma's niet kunnen worden geladen. In bijna alle gevallen kan dit euvel worden verholpen door de stand van de opname/weergavekop te corrigeren. Vrijwel alle draagbare cassetterecorders hebben een klein gat boven de kop dat toegang geeft tot een afregelschroefje. Soms is dit gat pas te vinden als de afsluitklep open staat. Met een kleine schroevedraaier kan het stelschroefje iets worden verdraaid. Dit afregelen kan het best gebeuren terwijl er een normaal voorbespeeld audiobandje draait dat bij voorkeur afkomstig is van een bekend platenlabel. Bij zo'n bandje mogen we ervan uitgaan dat de opname met een goed ingestelde kop is gemaakt. De schroef wordt zodanig afgesteld dat er zoveel mogelijk hoge tonen te horen zijn.

2 Pareltjes

Een parel is een kostbaar kleinood dat de moeite waard is om te bekijken. Een reeks parels echter, geregen tot een snoer, laat de schoonheid van dit natuurprodukt extra tot uiting komen.
Kleine programma's vertonen vaak een treffende gelijkenis met parels. Ze zijn niet zelden sterk beeldscherm-georiënteerd. Het visuele aspect is daardoor erg belangrijk. Een verrassend effect in kleur spreekt sterk tot de verbeelding. Een reeks kleinere programma's kan, samengevoegd tot een geheel, het idee van een parelsnoer weerspiegelen. Het volgende voorbeeld illustreert het parel-idee. Een miniatuurprogramma dat op eenvoudige wijze een mooi plaatje maakt.

```
10 REM parel
20 SCREEN 2:COLOR 15,4,5:CLS
30 CIRCLE(128,96),90,14,,,1.35
40 PAINT(128,96),14
50 GOTO 50
```

Het scherm wordt met SCREEN 2 geschikt gemaakt voor grafisch werken. Kleuren worden in MSX-BASIC aangegeven met een nummer.

0 Transparant (de kleur van de achtergrond)
1 Zwart
2 Groen
3 Lichtgroen
4 Donkerblauw (MSX meldt zich in deze kleur)
5 Lichtblauw
6 Donkerrood
7 Cyaan (heel licht paars-blauw)
8 Rood
9 Lichtrood
10 Donkergeel
11 Lichtgeel
12 Donkergroen
13 Magenta (violet)
14 Grijs (iets onderdrukt wit)
15 Wit (stralend wit)

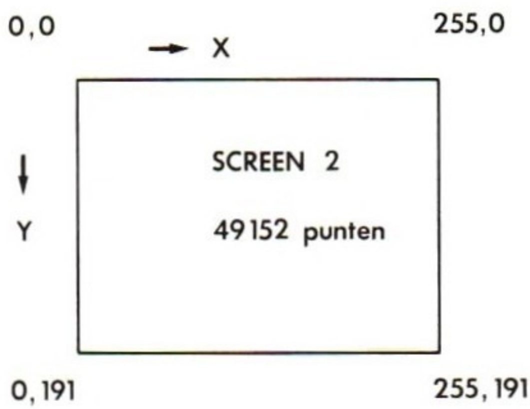
Het lijkt heel wat, maar als we de lijst nauwkeurig bekijken, moeten we constateren dat het eigenlijk tegenvalt:

Zwart
Wit   in twee helderheidsgraden.
Geel   in twee helderheidsgraden die erg dicht bij elkaar liggen.
Blauw   in twee helderheidsgraden.
Rood   in drie helderheidsgraden.

En dan de mengkleuren:

Groen   in drie helderheidsgraden.
Cyaan
Violet

In regel 20 van het programma is de inktkleur dus wit (14), de achtergrond donkerblauw (4) en de schermrand lichtblauw (5).
Met CLS wordt het scherm gewist en de kleurenopdracht uitgevoerd.
Als de opdracht SCREEN 2 wordt gegeven, is het MSX-beeld opgebouwd uit bijna 50.000 beeldpunten. Er zijn 192 horizontale rijen van 256 punten (zie afbeelding 2-1).



Afbeelding 2-1 De opbouw van scherm 2.

De coördinaten (128,96) geven het midden van het schermveld aan. Vanuit dit midden trekt regel 30 een cirkel met een straal van 90 beeldpunten en een grijze lijn (kleur 14). De begin- en de eindhoek van de cirkel worden niet opgegeven (,,,) zodat er een volledig gesloten cirkel ontstaat. De CIRCLE-opdracht met al zijn varianten krijgt in een later stadium onze uitgebreide aandacht.
Het getal 1.35 geeft de rondheid aan. Hoewel de meeste handleidingen aangeven dat er een zuiver ronde cirkel ontstaat als dit getal wordt weggelaten of als het getal 1 is, geeft de door de auteur gebruikte MSX-computer een zuiver ronde cirkel bij het getal 1.35. Mogelijk is dit niet voor alle machines gelijk.
Regel 40 vult het cirkelvlak grijs in.
Bij het gebruik van PAINT is het noodzakelijk dat de begrenzing dezelfde kleur heeft als de verfkleur. Is dit niet het geval dan loopt de verfkleur over.

Regel 50 zorgt ervoor dat het beeld netjes blijft staan als het programma afgelopen is. Laat hem maar eens weg! De parel krijgt al vorm.
Toch kan een contourlijntje hem nog aardig oppoetsen. En tenslotte maken we er een gaatje in:

```
42 CIRCLE(128,96),70,15,5,5.5,1.35
44 CIRCLE(128,96),70,15,5.6,6,1.35
46 CIRCLE(128,96),50,15,5,5.5,1.35
48 CIRCLE(128,96),50,15,5.6,6,1.35
49 CIRCLE(110,80),8,1,,,1.35
```

Geef de RUN-opdracht (F5) en ziedaar... ons eerste pareltje.
Als u een hekel heeft aan de nogal slordige regelnummering die nu is ontstaan, dan typt u gewoon RENUM in. Als het goed is, verschijnt de list zó op het scherm:

```
10  REM parel
20  SCREEN 2:COLOR 15,4,5:CLS
30  CIRCLE(128,96),90,14,,,1.35
40  PAINT(128,96),14
50  CIRCLE(128,96),70,15,5,5.5,1.35
60  CIRCLE(128,96),70,15,5.6,6,1.35
70  CIRCLE(128,96),50,15,5,5.5,1.35
80  CIRCLE(128,96),50,15,5.6,6,1.35
90  CIRCLE(110,80),8,1,,,1.35
100 GOTO 100
```

Toch best nog ingewikkeld met al die cijfertjes. Vooral het vastleggen van de begin- en eindhoek van een cirkelboog zoals dat in de regels 50 tot en met 80 gebeurd is, zal niet voor iedereen erg doorzichtig zijn. In een later stadium komen we hier echter uitgebreid op terug.
Het volgende programma laat alvast zien dat in een CIRCLE-opdracht niet altijd vaste punten behoeven te worden opgenomen. Afhankelijk van de waarde van I worden steeds nieuwe coördinaten uitgerekend.
Het programma is symbolisch voor het idee achter dit boek. Vele kleintjes kunnen tot een aantrekkelijk geheel worden samengevoegd.

```
10  REM pareltjes
20  SCREEN 2:COLOR 15,4,5:CLS
30  FOR I=1 TO 8
40    CIRCLE(128+15*I,100-I*I),8,,,,1.35
50    PAINT(126+15*I,100-I*I)
60    CIRCLE(128-15*I,100-I*I),8,,,,1.35
70    PAINT(126-15*I,100-I*I)
80  NEXT I
90  CIRCLE(128,100),8,,,,1.35
100 PAINT(126,100)
110 GOTO 110
```

De regels 40 en 50 worden herhaald in de regels 60 en 70. Eerst worden alle parels in de rechter beeldhelft getekend waarna de tekenprocedure wordt herhaald voor de linker beeldhelft. Dit komt tot uiting in de + of de − in de X-coördinaat.

Een parelsnoer... of toch niet. Bij nadere bestudering van het scherm blijkt dat de rijgdraad nog ontbreekt. Dit belangrijke onderdeel van het sieraad komt in het volgende hoofdstuk aan de orde.

3 De rijgdraad

Uit de handleiding van de computer blijkt dat er verschillende manieren bestaan om een programma op een cassettebandje te zetten. Bij elke methode van wegschrijven hoort een overeenkomstige manier om het programma van de band weer in het computergeheugen te laden. Er zijn drie van die combinaties met elk een eigen functie.

```
SAVE    LOAD
CSAVE   CLOAD
BSAVE   BLOAD
```

SAVE schrijft een BASIC-programma naar de band, precies zoals het op het scherm wordt weergegeven. Alle tekens worden netjes overgezet in ASCII-code. Deze code is internationaal vastgelegd.
LOAD haalt het programma terug in het geheugen, ook weer in ASCII-code. Bij het gebruik van SAVE en LOAD wordt het programma als een stuk tekst behandeld. Omdat BASIC-opdrachten altijd hetzelfde zijn, is het niet moeilijk een sterk verkorte code te verzinnen voor alle bestaande BASIC-opdrachten.
CSAVE werkt met zo'n sterk verkorte code en is voor het opslaan van BASIC-programma's dan ook het meest geschikt. Het BASIC-programma wordt door het gebruik van codes een flink stuk ingekort. Door te laden met CLOAD worden de ingekorte codes weer op de normale, voor ons leesbare wijze weergegeven.
BSAVE is uitsluitend bedoeld om programma's geschreven in machinetaal op te slaan. Deze methode is dan ook voorbehouden aan de specialist. In dit boek zult u daarom geen enkele machine-instructie tegenkomen. BLOAD laadt een machinetaalprogramma weer in het geheugen.
Bij SAVE- en LOAD-opdrachten hoort meestal een toevoeging:

```
SAVE"CAS:tekst"      LOAD"CAS:tekst"

BSAVE"CAS:ZAXXON"    BLOAD"CAS:ZAXXON"
```

De toevoeging CAS: geeft aan dat er met de cassette wordt gecommuniceerd. In de CSAVE- en CLOAD-opdracht wordt dat al door de C aangegeven. Uitsluitend de BLOAD-opdracht heeft de mogelijkheid er nog een extra opdracht aan toe te voegen. Althans volgens het handboek.

BLOAD"CAS:ZAXXON",r laadt het machinetaalprogramma ZAXXON van de cassette en start dit automatisch! De 'r' staat dan ook voor RUN. Alleen machinetaalprogramma's kunnen zelfstartend gemaakt worden.

Pareltjes zijn korte programma's die elkaar oproepen. Om pareltjes aan elkaar te kunnen rijgen, moet er iets verzonnen worden om ook gewone BASIC-programma's zelfstartend te maken. We proberen dus wat en schrijven een kort programma:

```
10 CLS:PRINT"Autostart test"
```

We saven het programma op de voorgeschreven manier met CSAVE"test". Let hierbij op het volgende: "Test" is voor de MSX-computer iets anders dan "test" of "TEST". In programmanamen wordt er onderscheid gemaakt tussen hoofdletters en kleine letters. Het is een goede gewoonte om voor programmanamen altijd kleine letters te gebruiken.
We doen net of we de handleiding niet gelezen hebben en proberen tegen beter weten in de autostarttoevoeging 'r'.
Spoel het bandje terug en probeer:

```
CLOAD"test",r
```

... jammer, dat werkt dus niet. Moeten alle zelfstartende programma's dan toch in machinetaal worden ontwikkeld? Alleen de gedachte al is zó onaantrekkelijk dat we eerst nog iets anders proberen.

```
10 CLS:PRINT"Autostart test"
```

We schrijven het BASIC-programma weg als een normale tekst. Dus in de onverkorte ASCII-vorm met behulp van:

```
SAVE"test"
```

Als het goed is, gaat de recorder lopen. Bij langere programma's is merkbaar dat het wegschrijven meer tijd kost dan wanneer met CSAVE zou worden weggeschreven. In het kader van onze bedoeling (pareltjes aan een snoertje) levert dat echter geen bezwaar op.
Laden proberen we met de bijbehorende opdracht:

`LOAD"test",r` (dus met de autostart-toevoeging)

Dat werkt dus. De tape gaat lopen. Het programma wordt ingelezen. Het scherm wordt gewist en daar verschijnt de tekst 'Autostart test'.
Het begin is er. Hoewel dit al aardig is, zijn we er nog niet helemaal. Het is immers onze bedoeling om programma's aan elkaar te rijgen. Een lopend programma moet zelfstandig een volgend programma van de band af laden. Dat programma moet zichzelf starten en daarna op zijn beurt weer het volgende programma oproepen enz. Om te onderzoeken of dat mogelijk is, maken we twee korte programma's.

Eén hebben we al bijna af:

```
10 REM test1
20 CLS:PRINT"Autostart test"
30 LOAD"test2",R
```

Dit programma wordt op tape gezet met:

```
SAVE"test1"
```

Het tweede programma luidt:

```
10 REM test2
20 SCREEN 2:COLOR 15,4,4:CLS
30 CIRCLE (130,100),80
40 GOTO 40
```

Ook dit programma wordt op de band gezet, in dit geval met:

```
SAVE"test2"
```

Nu komt de klap op de vuurpijl. De band wordt teruggespoeld tot voor het eerste programma. We typen de opdracht:

```
LOAD"test1",r
```

De recorder gaat lopen. Het scherm wordt gewist. De tekst 'Autostart test' verschijnt in beeld. De recorder start nogmaals. Het scherm wordt weer gewist en er verschijnt een witte cirkel in een blauw veld. Het kan dus. We kunnen nu elk BASIC-programma geschikt maken om een volgend programma van de band in het geheugen te laden en uit te laten voeren.
Het rijgsnoer is klaar... of kan het nog mooier?
Toch wel:

RUN"test2" voldoet ook als laadopdracht.

Vooropgesteld dat het programma met SAVE is opgeslagen. Een kleine handicap is dat de naam van het volgende programma bekend moet zijn op het ogenblik dat we de RUN-opdracht in de laatste programmaregel opnemen. Natuurlijk kunnen we alle programma's eenzelfde naam geven. Eenvoudiger is de volgende truc:

```
RUN"CAS:"
```

Deze opdracht kijkt naar het volgende programma op de band, ongeacht de naam en maakt het zelf-startend.

We kennen nu drie manieren om een programma zelf zijn opvolger tot actie aan te sporen.

```
LOAD"naam",r
RUN"naam"
RUN"CAS:"
```

In alle gevallen moet het programma met

```
SAVE"naam"
```

op de band zijn gezet.
We gebruiken de eenvoudigste:

```
RUN"CAS:"
```

We gaan aan de slag. Binnen een uur kunt u al een aardige show op het scherm presenteren.

4 Showtime

Een show begint met een entree. Toeters, bellen en glitter. Het onderstaande programma brengt dat op bescheiden wijze tot uiting.

```
10  REM entree
20  SCREEN 3:COLOR 10,4,4:CLS
30  PLAY"T3205E","T3204A","T3206E"
40  OPEN"GRP:" FOR OUTPUT AS #1
50  PSET(80,30),4
60  PRINT#1,"MSX"
70  PSET(65,90),4
80  COLOR 13
90  PRINT#1,"SHOW"
100 PSET(65,160),4
110 COLOR 3
120 PRINT#1,"TIME"
130 RUN"CAS:"         (130 GOTO 130)
```

Bij het intypen van de programma's uit dit boek, maar ook bij het zelf ontwikkelen van zelfstartende programma's, is het handig om de laatste regel pas toe te voegen als het programma goed werkt. In plaats van het volgende programma op te starten, kunnen we het programma ook laten pauzeren door in de laatste regel een oneindige lus op te nemen. (In de programmalistings staan beide mogelijkheden. Typ dus één van beide in.) Dit om te voorkomen dat de recordermotor per ongeluk wordt gestart als het programma tijdens proefdraaien de laatste regel bereikt. Er staat namelijk nog geen volgend programma op de band. Bij het rijgen van een parelsnoer zijn we altijd met de laatste parel bezig. Het is natuurlijk ook mogelijk de recorder nog niet op PLAY (PLAY/LOAD bij DATA-recorders) te zetten.
Als het programma zonder fouten in het geheugen staat, moet het op de cassette worden gezet:

```
SAVE"entree"
```

Het losse pareltje dat we al eerder gemaakt hebben zetten we er direct achter, nu uiteraard met de intussen welbekende laatste regel.

```
10  REM parel
20  SCREEN 2:COLOR 15,4,5:CLS
30  CIRCLE(128,96),90,14,,,1.35
40  PAINT(128,96),14
50  CIRCLE(128,96),70,15,5,5.5,1.35
60  CIRCLE(128,96),70,15,5.6,6,1.35
70  CIRCLE(128,96),50,15,5,5.5,1.35
```

```
80  CIRCLE(128,96),50,15,5.6,6,1.35
90  CIRCLE(110,80),8,1,,,1.35
100 RUN"CAS:"         (100 GOTO 100)
```

Vergeet niet bij het wegschrijven naar tape de recorder op RECORD (REC/SAVE) te zetten.
Tussen de programma's wordt geen extra pauze opgenomen. Ze komen direct achter elkaar. Als er een nieuw pareltje klaar is, wordt de opnametoets ingedrukt. De SAVE-opdracht start dan vanzelf de recorder.
Het allerlaatste programma aan het parelsnoer zal de recorder starten op zoek naar een opvolger. Als niemand ingrijpt, loopt het bandje gewoon verder. Niets aan de hand dus, want aan het eind slaat de recorder netjes af.
Voor we ons wat systematischer met pareltjes gaan bezighouden, voegen we – om de vaart erin te houden – eerst nog een ander, al eerder beschreven miniprogramma toe.
Na de MSX-entree en de losse parel kunnen we toch niet zonder een afbeelding van een compleet snoer. Toen we dit programma eerder behandelden, waren we de rijgtechniek nog niet meester. Nu kunnen we gaatjes boren en aanrijgen.

```
10  REM pareltjes
20  SCREEN 2:COLOR 15,4,5:CLS
30  FOR I=1 TO 8
40    CIRCLE(128+15*I,100-I*I),8,,,,1.35
50    PAINT(126+15*I,100-I*I)
60    CIRCLE(128-15*I,100-I*I),8,,,,1.35
70    PAINT(126-15*I,100-I*I)
80  NEXT I
90  RUN"CAS:"                  (90 GOTO 90)
```

Stoort u zich aan het feit dat het rijgsnoer zelf nog steeds ontbreekt? Wat let u er zelf een te programmeren. Tip: maak een witte cirkel met een grote straal. Leg het middelpunt ergens in het midden boven het scherm. Dit wordt bereikt door een negatieve Y-coördinaat te kiezen.
Zoals reeds eerder gesteld, is dit boek geen cursus in programmeertechnieken. Toch zullen we bij het spelen met de MSX-computer zeer veel aspecten tegenkomen. Om het terugzoeken van onderwerpen in dit boek een beetje gemakkelijk te maken, zullen we enige lijn proberen aan te brengen. De technieken zullen enigszins worden gegroepeerd, zonder de grenzen erg scherp te leggen. Het spelelement blijft voorop staan. Mocht u niet geïnteresseerd zijn in de opbouw van de programma's dan kan de tussen de programma's opgenomen tekst natuurlijk gewoon worden overgeslagen. Het is best leuk om de parels alleen maar in te typen en aan elkaar te rijgen. Miniprogramma's uit tijdschriften kunnen simpel worden toegevoegd.

5 Tekenen op het tekstscherm

Als de computer wordt aangezet, is scherm 0 te zien. Op dit scherm van 24 lijnen met 40 tekenposities kunnen geen grafische opdrachten uitgevoerd worden. Datzelfde geldt voor scherm 1, dat 24 regels heeft met 32 iets grotere tekenposities. Toch wil dat niet zeggen dat op deze tekstschermen geen mooie plaatjes kunnen worden gemaakt. De MSX-computer beschikt immers over een zeer uitgebreide reeks voorgeprogrammeerde figuurtjes. Een aantal is via het toetsenbord op te roepen met behulp van de GRAPH-toets, bij een deel ervan dient ook de SHIFT-toets te worden ingedrukt. De rest kan met de CODE- of SHIFT/CODE-toets worden opgeroepen.

We proberen het gebruik van de ingebouwde grafische tekenset even:

```
10 REM graphics op scherm 0
20 SCREEN 0:COLOR 15,4,4:CLS
30 KEY OFF:WIDTH 40
40 FOR I=1 TO 480
50   PRINT"▨▧";
60 NEXT I
70 LOCATE 16,11
80 PRINT"☺☺☺☺☺☺☺☺"
90 RUN"CAS:"                    (90 GOTO 90)
```

Regel 50: [GRAPH][Q] en [GRAPH][q]
Regel 80: acht keer [GRAPH][[]

Dat valt tegen. De figuurtjes overlappen elkaar een stukje. Dat is erg goed te zien bij het smile-gezichtje. Een verklaring hiervoor hoeft niet ver te worden gezocht. Op scherm 0 heeft elk teken een breedte van 6 puntjes. De grafische tekens zijn echter ontworpen met een breedte van 8 puntjes. Scherm 0 is dus echt niet geschikt voor grafisch werk. Het bovenstaande programma is gemakkelijk om te bouwen voor scherm 1.

```
10 REM graphics op scherm 1
20 SCREEN 1:COLOR 15,4,4:CLS
30 KEY OFF:WIDTH 32
40 FOR I=1 TO 384
50   PRINT"▨▧";
60 NEXT I
70 LOCATE 12,11
80 PRINT"☺☺☺☺☺☺☺☺"
90 RUN"CAS:"      (90 GOTO 90)
```

We zien nu een prachtig aaneengesloten visgraatmotief. Ook de smile-gezichtjes zijn helemaal afgebeeld.
Scherm 1 leent zich dus best wel voor grafisch werk. Er kunnen zelfs tekenfilmpjes op worden gemaakt. In een paar stappen zullen we een tekenfilmpje opbouwen. We beginnen met de hoofdrolspeler:

```
10 REM eend
20 SCREEN 1:CLS:KEY OFF
30 WIDTH 32
40 PRINT"    □<
50 PRINT"▀█▒█▀
```

Regel 40: *vier keer [spatie], [GRAPH][J],[<]*
Regel 50: *[GRAPH][O],[GRAPH][P],[GRAPH][q],[GRAPH][P],[GRAPH][O]*

Eendje, ga je mee? We geven het diertje een ander plaatsje op het scherm:

```
10 REM eend
20 SCREEN 1:CLS:KEY OFF
30 WIDTH 32
40 X=10:Y=10
50 LOCATE X,Y:PRINT"    □<
60 LOCATE X,Y+1:PRINT"▀█▒█▀
```

Bewegen is nu ook niet moeilijk meer. We passen regel 40 aan en voegen twee regels toe:

```
10 REM turbo eend
20 SCREEN 1:CLS:KEY OFF
30 WIDTH 32
32 Y=2
40 FOR X=1 TO 32
50   LOCATE X,Y:PRINT"    □<
60   LOCATE X,Y+1:PRINT"▀█▒█▀
70 NEXT X
```

RUN

Dat gaat hard. Heeft u al eens eerder een eend met raketvoortstuwing gezien? Om het spoor te wissen, wordt er een extra spatie opgenomen als eerste teken van de twee PRINT-opdrachten.

```
50   LOCATE X,Y:PRINT"     □<
60   LOCATE X,Y+1:PRINT" ▀█▒█▀
```

Dat ziet er al een stuk beter uit. Maar het lijkt nog steeds meer op een kanonskogel dan op een levend diertje. We leren hem vliegen.

```
10 REM turbo eend met vleugels
20 SCREEN 1:CLS:KEY OFF
30 WIDTH 32
32 Y=2
40 FOR X=1 TO 32 STEP 2
50 LOCATE X,Y:PRINT"     ◘<
60 LOCATE X,Y+1:PRINT"▬█▨█▬
70 LOCATE X+1,Y:PRINT"   ▧ ◘<
80 LOCATE X+1,Y+1:PRINT"▬███▬
90 NEXT X
```

Regel 70: drie keer [spatie],[GRAPH][Q],[spatie],[GRAPH[J],[<]

RUN

De regels 50 en 60 zijn met een kleine verandering herhaald in 70 en 80. De verandering zit niet alleen in de verhoging van de waarde X met 1. Het eendje heeft bovendien een vleugeltje gekregen. Ook regel 40 heeft een kleine uitbreiding ondergaan. Hij vliegt, maar ach jee wat een haast! Twee extra regeltjes zorgen ervoor dat we de vliegsnelheid beheersen.

```
62 FOR Q=1 TO 100:NEXT Q
82 FOR Q=1 TO 100:NEXT Q
```

Deze lus wordt in programma's vaak gebruikt om wat tijd te rekken. Het getal geeft aan hoeveel keer de computer in deze lus rond moet rennen voor hij verder mag. Vul er maar eens wat anders voor in.
Wat dacht u van een slootkant?

```
33 FOR I=0 TO 10
34   LOCATE I,16:PRINT"*"
35   LOCATE I,17:PRINT"+"
36   LOCATE I,18:PRINT"⊥"
37 NEXT I
38 LOCATE 0,19:PRINT STRING$(32,215)
```

Regel 34: [GRAPH][Z]
Regel 35: [SHIFT][GRAPH][G]
Regel 36: [GRAPH][B]

Nu komt het probleem van scherm 1 aan het licht. Het is zonder kunstgrepen niet mogelijk om meer kleuren af te beelden. Bij gebruik van de opdracht COLOR 10 worden zowel de eend als de slootkant geel. Deze beperking doet ons ertoe besluiten de tekstschermen 0 en 1 verder links te laten liggen. Echter niet voordat het werk dat we in dit hoofdstuk gedaan hebben als een echt pareltje aan de ketting is geregen. Eerst hernummeren we het programma met RENUM, dan voegen we de kettingaansluiting toe. Als alles goed gegaan is, ziet het er ongeveer zó uit:

```
10  REM turbo eend met vleugels
20  SCREEN 1:COLOR 15,4,4:CLS:KEY OFF
30  WIDTH 32
40  Y=2
50  FOR I=0 TO 10
60    LOCATE I,16:PRINT"*"
70    LOCATE I,17:PRINT"+"
80    LOCATE I,18:PRINT"⊥"
90  NEXT I
100 LOCATE 0,19:PRINT STRING$(32,215)
110 FOR X=1 TO 32 STEP 2
120   LOCATE X,Y:PRINT"      ◘<"
130   LOCATEX,Y+1:PRINT"▬█▒█▬ "
140   FOR Q=1 TO 100:NEXT Q
150   LOCATE X+1,Y:PRINT"     ▒ ◘<
160   LOCATE X+1,Y+1:PRINT"▬███▬ "
170   FOR Q=1 TO 100:NEXT Q
180 NEXT X
190 RUN"CAS:"              (190 GOTO 190)
```

Het zal duidelijk zijn dat de techniek die toegepast is om het eendje te laten vliegen ook benut kan worden om teksten over het scherm te laten bewegen. Maak zelf eens een bewegende lichtkrant en rijg hem aan de ketting met:

```
RUN"CAS:"
```

6 Scherm 3

Hoewel de term 'lage resolutie' doet vermoeden dat er nogal wat beperkingen aan scherm 3 kleven, pakt de praktijk anders uit. Geringschattende opmerkingen over de slechte kwaliteit van de op dit scherm weer te geven beelden komen voort uit onbekendheid of uit gebrek aan fantasie. In tegenstelling tot de tekstschermen, waarop slechts in één kleur kan worden gewerkt en het hoge-resolutiescherm dat later aan de orde komt, kan bij scherm 3 ieder beeldpunt elke kleur aannemen die de MSX-computer tot zijn beschikking heeft. Goed, de beeldpunten zijn wat groot uitgevallen, maar voor sommige beelden is dat juist prachtig. Het geeft de plaatjes een echt computeraspect. Kijk maar:

```
10  REM msx
20  SCREEN 3,3:COLOR 15,1,1:CLS
30  FOR I=0 TO 2
40    I$=""
50    FOR S=0 TO 31
60      READ S$
70      I$=I$+CHR$(VAL("&H"+S$))
80    NEXT S
90    SPRITE$(I)=I$
100 NEXT I
110 DATA 00,00,00,00,00,00,00,00,00,00,
         00,00,00,01,01,00
120 DATA 00,00,00,00,39,39,7D,7F,7F,7F,
         FF,FF,EF,EF,C7,00
130 DATA 00,00,00,00,C7,CF,CF,EE,EF,EF,
         F7,F0,7F,7F,3F,00
140 DATA 00,00,00,00,FE,FF,FF,03,F1,F8,
         F9,3B,FF,FF,FF,00
150 DATA 00,00,00,00,07,0F,9E,FC,F8,F0,
         F8,FC,DE,8F,07,00
160 DATA 00,00,00,00,80,00,00,00,00,00,
         00,00,00,00,00,00
170 FOR X=1 TO 80
180   Y=X*4/5
190   PUT SPRITE 0,(X,Y),8,0
200   PUT SPRITE 1,(X+32,Y),8,1
210   PUT SPRITE 2,(X+64,Y),8,2
220 NEXT X
230 CIRCLE(128,85),40,10
240 COLOR,,4
250 OPEN"GRP:" FOR OUTPUT AS #1
260 PSET(7,150),1
270 COLOR 12
280 PRINT#1,"COMPUTER"
290 RUN"CAS:"               (290 GOTO 290)
```

Het programma loopt vooruit op een aantal zaken die nog uitvoerig aan de orde zullen komen, zoals het maken van sprites, tekenen van geometrische figuren en teksten op het grafische scherm. Het programma illustreert dat afbeeldingen op het lageresolutiescherm best wel aantrekkelijk kunnen zijn. De ketting bevat nu al een aantal pareltjes. Spoel het bandje maar eens helemaal terug en start het met :

```
RUN"CAS:"
```

En dit is nog maar het begin.
Een volgend pareltje laat alle kleuren zien die de MSX-computer kan weergeven. Een klein programma met een bont resultaat.

```
10 REM ruit
20 SCREEN 3:CLS
30 FOR X=0 TO 256 STEP 4
40   FOR Y=0 TO 192 STEP 4
50      C=C+1
60      IF C=16 THEN C=0
70      PSET(X,Y),C
80 NEXT Y,X
90 RUN"CAS:"  (90 GOTO 90)
```

Uit dit programma blijkt duidelijk hoe de beeldopbouw op het lageresolutiescherm plaatsvindt. Het scherm kent even veel beeldpunten als het hogeresolutiescherm 2. Namelijk 256 × 192. Echter, als een enkel beeldpunt wordt aangewezen, dan maakt de computer een blokje van 4 × 4 beeldpunten. In de regels 30 en 40 wordt dan ook horizontaal en verticaal met stapjes van vier beeldpunten gesprongen. Regel 70 doet het eigenlijke werk. Hij wijst het punt aan waar het blokje komt dat moet worden gekleurd. De kleur wordt in regel 50 steeds veranderd. Omdat kleurnummer 16 niet bestaat, laat regel 60 de kleurtelling steeds opnieuw beginnen. Een aardig effect, nietwaar? Een variant wordt in het volgende stukje beschreven.

```
10 REM lapjesdoek
20 SCREEN 3:COLOR ,1,1:CLS
30 FOR X=0 TO 256 STEP 4
40   FOR Y=0 TO 192 STEP 4
50      C=RND(1)*16
60      PSET(X,Y),C
70 NEXT Y,X
80 RUN"CAS:"  (80 GOTO 80)
```

In MSX-BASIC mogen niet-gebruikte variabelen gewoon worden weggelaten. Zo ontbreekt het eerste getal in de COLOR-opdracht. Deze inktkleur wordt immers apart vastgelegd als er in regel 60 een punt wordt afgebeeld. In tegenstelling tot het vorige programma wordt de inktkleur door regel 50 willekeurig bepaald. Maar waarom staat er 16, er zijn toch maar 15 kleuren? RND(1) levert een getal tussen 0 en 1. De 1 komt nooit voor. Het maximale getal voor C kan

daarom ook geen 16 worden, maar hoogstens 15,999999. In PSET wordt het getal afgebroken zodat er netjes 15 overblijft. Vergeet regel 80 niet!
Teksten kunnen niet zonder meer met de gebruikelijke PRINT-opdracht op een grafisch scherm worden gezet. Probeer het maar met het volgende programma:

```
10 SCREEN 3:CLS
20 PRINT "Dit is een tekst"
```

```
RUN
```

De foutmelding geeft aan dat er iets niet goed zit. De handleiding vertelt dan ook dat er geen tekens worden geaccepteerd op de grafische schermen 2 en 3. Dat lijkt een geweldig grote handicap. In veel figuren willen we immers ook met teksten werken. MSX-BASIC heeft echter een zeer flexibele mogelijkheid om teksten in elke gewenste kleur op elke gewenste plaats op het scherm te zetten. We hoeven hiervoor slechts een kanaal naar het grafische scherm te openen. De MSX-computer verlangt wel dat we zo'n kanaal een eigen nummer geven. Bijvoorbeeld: open kanaal nummer 1 voor uitvoer naar het scherm. Of, in echt BASIC:

```
OPEN "GRP:" FOR OUTPUT AS #1
```

Hiermee wordt het bovenstaande mislukte programma aangepast.

```
10 SCREEN 3:CLS
20 OPEN "GRP:" FOR OUTPUT AS #1
30 PRINT #1,"Dit is een tekst"
```

```
RUN
```

In de PRINT-opdracht wordt tot uiting gebracht dat de te printen tekst voor kanaal 1 bedoeld is.
Zo, het probleem lijkt opgelost. Hoewel, zijn de letters niet wat groot uitgevallen? Bij SCREEN 3 worden alle tekens vier keer zo groot dan normaal. Grote letters zijn niet gek voor reclameboodschappen. Het logo van een bekend Amerikaans bedrijf bijvoorbeeld.

```
10 REM logo
20 SCREEN 3:COLOR 10,4,4:CLS
30 OPEN "GRP:" FOR OUTPUT AS #1
40 PSET(35,77),4
50 PRINT#1,"DuPont"
60 CIRCLE(124,90),110,,,,.5
70 RUN"CAS:"          (70 GOTO 70)
```

Let erop hoe met een PSET-opdracht de plaats wordt aangegeven waar de tekst moet komen. Dit gaat met beeldpuntnauwkeurigheid, maar ook hier

weer afgerond naar blokjes van 4 × 4. De kleur in de PSET-opdracht (4) is gelijk gekozen aan de achtergrondkleur. Onzichtbaar dus. De CIRCLE-opdracht komt verderop uitgebreid aan de orde. Vergeet geen komma's en punten. Spaties daarentegen zijn minder belangrijk.
Van de grote letters kunnen we dankbaar gebruik maken. Een parelketting met reclameboodschappen of mededelingen is op meters afstand goed te lezen. Als de letters tenminste netjes gerangschikt op het scherm staan. Niet zó dus:

```
10  REM letters
20  SCREEN 3:COLOR ,1,1:CLS
30  A$="   MSX":B$="COMPUTERS"
40  OPEN "GRP:" FOR OUTPUT AS #1
50  FOR I=1 TO 100
60    COLOR RND(1)*16
70    PRINT#1,A$
80    PSET(RND(1)*250,RND(1)*180)
90    PRINT#1,B$
100 NEXT I
110 PSET(80,50):COLOR 1:PRINT#1,"■■■■"
120 PSET(80,75):COLOR 1:PRINT#1,"■■■■"
130 PSET(90,70):COLOR 10:PRINT#1,"MSX"
140 RUN"CAS:"             (140 GOTO 140)
```

Regel 110 en 120: vier keer [GRAPH][P]

Weer een pareltje aan de ketting. Tekens kunnen op alle gebruikelijke manieren, door een geopend kanaal, naar het grafische scherm worden gestuurd. Direct met PRINT#1," ... ". Maar ook indirect met A$=" ... " gevolgd door PRINT#1,A$. Ook ASCII-codes worden geaccepteerd, zoals PRINT#1,CHR$(177). Deze laatste manier is handig bij het gebruik van de voorgeprogrammeerde grafische tekens. In een afdruk van een programma is de ASCII-code altijd mogelijk, terwijl de meeste printers niet in staat zullen zijn om de grafische blokjes af te drukken.
Er schuilt echter een addertje onder het gras. De eerste 31 tekens kunnen niet met CHR$(...) worden opgeroepen. Het is zeker de moeite waard om het volgende programma even in te typen. Maak er meteen een pareltje van.

```
10  REM tekens op het scherm met CHR$()
20  SCREEN 3:COLOR 15,4,4:CLS
30  OPEN "GRP:" FOR OUTPUT AS #1
40  FOR I=0 TO 255
50    PSET(0,10)
60    PRINT#1,I;CHR$(I)
70    FOR Q=1 TO 200:NEXT Q
80    CLS
90  NEXT I
100 RUN"CAS:"             (100 GOTO 100)
```

Op het scherm verschijnen achtereenvolgens de ASCII-waarden van alle tekens die de MSX-computer kent. Dat zijn er op een paar na 256. Achter het

nummer wordt het desbetreffende teken afgebeeld. De eerste 31 tekens worden niet afgebeeld, dan volgt het teken voor de spatie CHR$(32), die is dus ook niet te zien, en vervolgens paradeert de gehele tekenset voorbij. Een voorstelling op zich, die weer eens laat zien hoe veelzijdig de MSX-computer is.
De snelheid is te regelen door regel 70 aan te passen.
Ter afsluiting van de reeks parels die we nu gemaakt hebben op het lageresolutiescherm voegen we er nog twee aan toe.

```
10  REM lijnenspel
20  SCREEN 3:COLOR ,1,1:CLS
30  FOR I=1 TO 255
40    COLOR RND(1)*14
50    LINE(0,0)-(RND(1)*256,RND(1)*192)
60    LINE(256,0)-(RND(1)*256,RND(1)*192)
70    LINE(0,192)-(RND(1)*256,RND(1)*192)
80    LINE(256,192)-(RND(1)*256,RND(1)*192)
90  NEXT I
100 RUN"CAS:"                    (100 GOTO 100)
```

Het getal 14 in regel 40 zet grijs (14) en wit (15) buitenspel.
Om op alles voorbereid te zijn, kan het volgende pareltje al worden ingetypt. Het opgeroepen beeld zal ons ongetwijfeld in de juiste sfeer brengen.

```
10  REM kerstboom
20  SCREEN 3:COLOR ,1,1:CLS
30  FOR I=1 TO 40
40    PSET(RND(1)*250,RND(1)*145),RND(1)*16
50  NEXT I
60  FOR I=1 TO 30
70    PSET(130-1.5*I,20+4*I),12
80    A$="R"+STR$(3*I)
90    DRAW A$
100 NEXT I
110 LINE(126,145)-(134,165),10,BF
120 PSET(110,165),12
130 DRAW"M150,165M140,192M120,192M110,165"
140 RUN"CAS:"                    (140 GOTO 140)
```

Genoeg over lage resolutie. Mogelijk heeft u inmiddels zelf wat pareltjes gecreëerd. Neem ze op in de groeiende reeks programmaatjes.

7 Scherm 2

Op scherm 2 kunnen alle registers open. Op uitbundige wijze kunnen we hierop stoeien met punten, lijnen, cirkels en vierkanten. Sprites komen op dit scherm het mooist tot hun recht. Spelenderwijs zullen we in dit deel van het boek leren omgaan met de veelzijdige grafische gereedschappen die de MSX-computer ons biedt. Maar ook de beperkingen komen aan de orde.
Hier volgt alvast een voorproefje om als parel aan het snoer te rijgen.

```
10  REM lijn
20  SCREEN 2:COLOR 6,1,1:CLS
30  FOR A=1 TO 8
40    CIRCLE(130,100),A
42    A$="O=A;L60CDEFGABAGFECD"
44    B$="L60DEFGABAGFEDCC"
46    C$="L60EFGABAGFEDCD"
50    PLAY A$,B$,C$
60  NEXT A
70  FOR I=1 TO 100
80    PSET(130,100)
90    LINE-(RND(1)*256,RND(1)*192),RND(1)*14
100 NEXT I
110 RUN"CAS:"                    (110 GOTO 110)
```

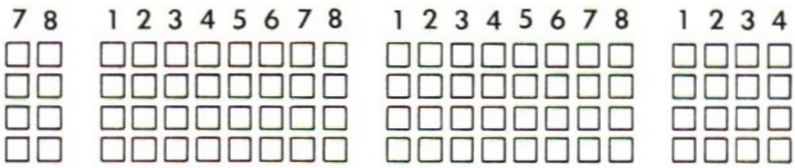
In dit programma komen elementen voor die in de volgende hoofdstukken verder zullen worden uitgewerkt. Maar ook andere grafische opdrachten komen aan de orde. De belangrijkste zijn:

```
PSET (PRESET)
LINE
DRAW
CIRCLE
PAINT
```

Voor we echter overgaan tot een meer uitgebreide toepassing van deze commando's zullen we eerst enkele eigenaardigheden van het hogeresolutiescherm bekijken. Uiteraard aan de hand van een nieuw pareltje aan de ketting.

```
10 REM probleempje
20 SCREEN 2:COLOR 1,15,15:CLS
30 FOR I=50 TO 200
40   C=C+1:IFC=16 THEN C=0
50   LINE(I,0)-(I,192),C
60   FOR Q=1 TO 300:NEXT Q
70 NEXT I
80 RUN"CAS:"      (80 GOTO 80)
```

Er gebeurt nu iets zeer merkwaardigs. Verticaal worden lijnen getrokken, steeds met een opvolgend kleurnummer. De reeds getrokken lijnen veranderen echter in de kleur van de laatst getrokken lijn. Na acht lijntjes blijft de kleur behouden en begint een volgende baan steeds van kleur te veranderen. In de praktijk zou dit betekenen dat er op de meeste plaatsen van het scherm geen twee verticale lijnen van verschillende kleur naast elkaar kunnen worden afgebeeld. Om dit probleem te kunnen omzeilen, bekijken we in afbeelding 7-1 de schermopbouw van heel dichtbij.



Afbeelding 7-1 Beeldpunten op het monitorscherm.

De beeldpunten zijn per horizontale lijn verdeeld in groepjes van acht. In één zo'n groepje op de horizontale lijnen mogen slechts twee kleuren voorkomen. Stel dat de beeldpunten 1 t/m 7 de achtergrondkleur blauw hebben en dat beeldpunt 8 rood is gekleurd. Als nu één van de blauwe beeldpunten geel wordt gemaakt, dan nemen alle beeldpunten van het desbetreffende groepje de laatst toegevoegde kleur aan: alle punten worden geel.
Willen we toch lijnen van verschillende kleur verticaal naast elkaar, dan kan dat alleen maar aan de randen van de groepjes. Bijvoorbeeld rood in beeldpunt 8 en groen in het ernaast liggende beeldpunt 1. De drie kleuren zijn dan netjes verdeeld over twee groepen.
Het volgende programma maakt gebruik van deze handigheid om ruimtelijke staafdiagrammen te maken. Het levert een mooi schermpje op.

```
10 REM foutje weg
20 SCREEN 2:COLOR 15,1,1:CLS
30 LINE(20,0)-(240,192),14,B
40 FOR I=1 TO 4
50   LINE(90-3*I,20+16*I)-(110-
     3*I,100+16*I),2*I+1,BF
60   LINE(216-16*I,20+16*I)-(240-
     16*I,100+16*I),2*I+1,BF
70 NEXT I
80 RUN"CAS:"        (80 GOTO 80)
```

De kolommen die door regel 50 worden getekend, zijn drie beeldpunten ten opzichte van elkaar verschoven. Binnen een groepje van acht horizontale punten kunnen dan ook meer dan twee kleuren voorkomen. De laatst ingegeven kleur overheerst, zoals aan het voorste rode en het op één na achterste blauwe vlak is te zien. Door het geheel iets te verschuiven kan misschien een punt worden gevonden waarbij de schoonheidsfout niet optreedt. Verreweg de handig-

ste methode om het probleem te omzeilen, wordt gedemonstreerd in regel 60. In deze regel is de afstand tussen de kolommen maar liefst 16 beeldpunten (acht zou genoeg zijn).
Eigenlijk is het wonderbaarlijk dat met zo'n eenvoudig programma al aardig complexe 'business-grafieken' kunnen worden gemaakt.
Het hier behandelde euvel speelt zich gelukkig alleen maar af langs horizontale groepjes beeldpunten. Verticaal mogen alle beeldpunten verschillend gekleurd zijn. Het volgende stukje BASIC illustreert dat.

```
10  REM groepjes van acht
20  SCREEN 2:COLOR ,1,1:CLS
30  FOR X=50 TO 70
40    FOR Y=50 TO 114
50      IF C=16 THEN C=0
60      PSET(X,Y),C
70      C=C+1
80  NEXT Y,X
90  FOR X=90 TO 110
100   FOR Y=51 TO 114
110     IF C=16 THEN C=0
120     PSET(X,Y),C
130     C=C+1
140 NEXT Y,X
150 RUN"CAS:" (150 GOTO 150)
```

In het linkerdeel van het scherm doet zich het verschijnsel voor dat verticaal alle kleuren wel worden afgebeeld, maar de afzonderlijke horizontale lijntjes verschieten steeds weer van kleur als er een ander kleurtje langs komt. In het rechterdeel liggen de kleuren op gelijke hoogte en ontstaat er een regelmatig regenboogpatroon.
Als u wat programmeerervaring heeft, dan bekruipt u bij het bekijken van het voorgaande programma ongetwijfeld een gevoel van afgrijzen: tweemaal exact dezelfde routine achter elkaar! Om u weer enigszins tot rust te brengen, volgt hieronder een opgepoetste versie, waarbij de te herhalen routine in een subroutine is ondergebracht.

```
10  REM groepjes van acht
20  SCREEN 2:COLOR ,1,1:CLS
30  FOR X=50 TO 70
40    FOR Y=50 TO 114
50      GOSUB 120
60  NEXT Y,X
70  FOR X=90 TO 110
80    FOR Y=51 TO 114
90      GOSUB 120
100 NEXT Y,X
110 RUN"CAS:" (110 GOTO 110)
120 IF C=16 THEN C=0
130 PSET (X,Y),C
140 C=C+1
150 RETURN
```

Tot besluit van dit hoofdstuk over het hogeresolutie scherm volgen nog twee kleine programma's om aan de ketting te rijgen.

```
10 REM lijnenspel
20 SCREEN 2:COLOR ,4,4:CLS
30 FOR I=1 TO 500
40   LINE(0,0)-(RND(1)*256,RND(1)*192),
     RND(1)*16
50 NEXT I
60 RUN"CAS:"                  (60 GOTO 60)
```

Let vooral eens op de linkerbovenhoek. Het verschijnsel van de acht bij elkaar horende beeldpunten is daar heel goed te zien.

```
10 REM vierkanten
20 SCREEN 2:COLOR ,4,4:CLS
30 FOR I=1 TO 200
40   LINE(0,0)-(RND(1)*256,RND(1)*192),
     RND(1)*16,B
50 NEXT I
60 RUN"CAS:"                  (60 GOTO 60)
```

Valt het verschil met het vorige programma op? Inderdaad, afgezien van een kleiner aantal herhalingen heeft de LINE-opdracht in regel 40 een extra toevoeging. In het volgende hoofdstuk zullen we de LINE-opdracht aan een nader onderzoek onderwerpen.

8 Lijnen en wat daarbij hoort

Het lijkt eenvoudig; met de LINE-opdracht kunnen lijnen worden getrokken. Mooie rechte lijnen, kijk maar:

```
10  REM horizon
20  SCREEN 2:COLOR ,1,1:CLS
30  FOR V=1 TO 6
40    LINE(128-V*8,64)-(120-V*20,191),3
50    LINE(128+V*8,64)-(120+V*20,191),3
60  NEXT V
70  FOR I=1 TO 7
80    READ A
90    LINE(178+I*8,64)-(255,A),3
100   LINE(82-I*8,64)-(0,A),3
110 NEXT I
120 LINE(128,64)-(122,192),3
130 FOR H=1 TO 12
140   LINE(0,63+H*H)-(256,63+H*H),3
150 NEXT H
160 DATA 170,142,121,104,90,78,70
170 GOTO 170
```

Dat levert een uitgesproken futuristisch landschap op. En dat met zo'n klein programma. We rijgen dit pareltje aan de ketting, omdat het in de toekomst zeker nog eens van pas zal komen. Het beeld vormt immers een ideaal decor voor een ruimtespel. Vindt u het nog wat statisch? Daar doen we wat aan. In het volgende programma zijn veel regels uit het vorige programma opgenomen. Als u het handig aanpakt, kunt u zich aardig wat typewerk besparen.

```
10  REM horizon 2
20  SCREEN 2:COLOR ,1,1:CLS
30  FOR V=1 TO 6
40    LINE(128-V*8,64)-(120-V*20,191),3
50    LINE(128+V*8,64)-(120+V*20,191),3
60  NEXT V
70  FOR I=1 TO 7:READ A
80    LINE(178+I*8,64)-(255,A),3
90    LINE(82-I*8,64)-(0,A),3
100 NEXT I
110 LINE(128,64)-(122,192),3
120 LINE(0,63)-(256,63),3
130 FOR T=1 TO 20
140   FOR H=1 TO 11
150     LINE(0,63+H*H)-(256,63+H*H),1
160   NEXT H
170   FOR I=1 TO 10
```

```
180       PSET(RND(1)*256,RND(1)*62),RND(1)*15
190     NEXT I
200     CIRCLE(200,30),T,10:PAINT(200,30+T),10
210     FOR H=1 TO 11
220       LINE(0,63+H*H)-(256,63+H*H),3
230     NEXT H
240     DATA 170,142,121,104,90,78,70
250     F=T/3+1:PLAY"O=F;CDEFG"
260 NEXT T
270 RUN"CAS:"                    (270 GOTO 270)
```

Dat LINE-opdrachten ijzersterk zijn, wordt door dit programma wel bewezen. Boven deze voorbijtrekkende asteroïde zou je zo een UFO kunnen verwachten. Het hoofdstuk dat sprites beschrijft, zal dat dan ook mogelijk maken.
De LINE-opdracht is zeer veelzijdig.

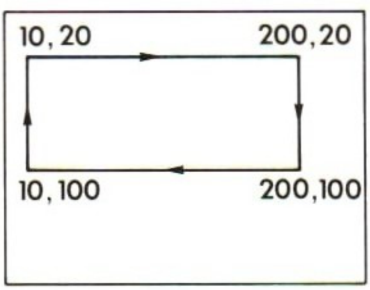
```
LINE(1Ø,2Ø)-(2ØØ,1ØØ),,B
```

trekt een lijn van beeldpunt (10,20) naar beeldpunt (200,100).

```
LINE(1Ø,2Ø)-(2ØØ,1ØØ),6
```

doet datzelfde in de kleur rood.
Met de LINE-opdracht zoals die hierboven staat, kunnen we een rode rechthoek (zie afbeelding 8-1) als volgt tekenen:

```
LINE(1Ø,2Ø)-(2ØØ,2Ø),6
LINE(2ØØ,2Ø)-(2ØØ,1ØØ),6
LINE(2ØØ,1ØØ)-(1Ø,1ØØ),6
LINE(1Ø,1ØØ)-(1Ø,2Ø),6
```



Afbeelding 8-1

Deze procedure is tamelijk omslachtig, vraagt veel typewerk en geeft dus veel kans op fouten.
De LINE-opdracht heeft een paar specialiteiten om het tekenen van rechthoeken te vereenvoudigen. Een doosje heet in het Engels 'box'. Door aan de LINE-opdracht de eerste letter van deze Engelse benaming (B) toe te voegen, hoe-

ven we slechts twee hoekpunten van een rechthoek op te geven om hem keurig op het scherm te krijgen:

```
LINE(10,20)-(200,100),6,B
```

Het kan natuurlijk voorkomen dat we een rechthoek in de op een bepaald ogenblik geldende inktkleur willen hebben. Het kleurcijfer mag dan vervallen, echter niet de bijbehorende komma!

```
LINE(10,20)-(200,100),,B
```

Deze regel geldt voor veel opdrachten waarin een aantal variabelen kan worden vastgelegd. Wordt in dit geval de komma per ongeluk weggelaten dan staat er

```
LINE(10,20)-(200,100),B
```

Dat is geen rechthoek, maar een diagonale lijn in de kleur zwart (B=0) of een andere kleur als de B in het programma al een andere waarde had.
MSX-BASIC kent de PAINT-opdracht. Bijvoorbeeld:

```
LINE(10,20)-(200,100),6,B:PAINT(50,50),6
```

Tussen de haakjes van de PAINT-opdracht wordt een punt opgegeven dat binnen de in te kleuren rechthoek valt. Bij het gebruik van PAINT wordt achter de haakjes het kleurnummer opgegeven. Dit moet dezelfde kleur zijn als die waarmee de rechthoek is getekend. Mocht dat niet het geval zijn, dan kookt de kleur over en vult het hele scherm.
Ook voor het inkleuren van rechthoeken heeft MSX-computer een foefje ingebouwd. De twee hiervoor afgedrukte opdrachten kunnen door een enkele worden vervangen:

```
LINE(10,20)-(200,100),6,BF
```

De toevoeging F is de eerste letter van het Engelse woord 'fill', dat vullen betekent.
Alvorens de LINE-opdracht verder te bekijken, rijgen we even twee pareltjes aan de ketting. Pareltjes die gebruik maken van de LINE-opdrachten zoals die hiervoor beschreven zijn.

```
10 REM rechthoeken
20 SCREEN 2:COLOR,1,1:CLS
30 FOR I=1 TO 100
40   LINE(0,0)-(RND(1)*256,RND(1)*192),
     RND(1)*16,BF
50   LINE(256,0)-(RND(1)*256,RND(1)*192),
     RND(1)*16,BF
```

```
60    LINE(256,192)-(RND(1)*256,RND(1)*192),
      RND(1)*16,BF
70    LINE(0,192)-(RND(1)*256,RND(1)*192),
      RND(1)*16,BF
80 NEXT I
90 RUN"CAS:"                   (90 GOTO 90)
```

Vanuit de vier hoeken worden in een razend tempo rechthoeken getekend van willekeurige afmetingen in een willekeurige kleur. Jammer dat Pieter Mondriaan met dit soort technieken al furore maakte, anders had er zeker brood in gezeten. Het volgende programma maakt de schilderijententoonstelling compleet. Hier wordt uitbundig gebruik gemaakt van de invulopdracht. De snelheid waarmee de computer zijn kleurboek vult, is bepaald indrukwekkend.

```
10 REM ingekleurd spectakel
20 SCREEN 2:COLOR ,4,4:CLS
30 FOR I=1 TO 100
40    LINE(RND(1)*256,RND(1)*192)-
      (RDN(1)*256,RND(1)*192),RND(1)*16,BF
50 NEXT I
60 RUN"CAS:"              (60 GOTO 60)
```

De LINE-opdracht kent nog meer mogelijkheden. Voor het trekken van een lijn vanaf het beeldpunt dat het laatst door de computer is getekend naar een ander punt, laten we het eerste deel van de opdracht weg.

```
LINE -(1ØØ,15Ø)
```

De mogelijkheid om met kleur gevulde vierkanten te maken, blijft bestaan.

```
LINE -(1ØØ,15Ø),1Ø,BF
```

Een voorbeeld waarin de verkorte LINE-opdracht is toegepast, wordt gegeven in het volgende programma. Door het gebruik van een goniometrische uitdrukking wordt een spiraal op het scherm getekend.

```
10  REM wentelgoot
20  SCREEN 2:COLOR 11,4,4:CLS
30  A=30:B=0:PI=3.14
40  PSET(180,0)
50  FOR T=0 TO 6*PI STEP PI/25
60    X=50*COS(T):G=126+X
70    A=A+1:B=B+1
80    LINE -(G,B)
90    LINE(126,A)-(G,B)
100 NEXT T
110 RUN"CAS:"   (110 GOTO 110)
```

In een ander voorbeeld wordt de verkorte LINE-opdracht gebruikt om een driehoek te tekenen.

```
10  REM driehoek met bal
20  SCREEN 2:COLOR 11,4,4:CLS
30  Y=192
40  FOR X=0 TO 192
50    PSET(0,0)
60    LINE-(X,Y)
70  NEXT X
80  FOR I=0 TO 190
90    CIRCLE(I,I-9),5,4
100   CIRCLE(I+1,I-8),5,15
110 NEXT I
120 RUN"CAS:"  (120 GOTO 120)
```

Als de bal langs de driehoek naar beneden rolt, geeft hij een prachtige demonstratie van het effect dat door de beeldpuntgroepjes van acht wordt veroorzaakt. Er is met het beeld nog iets vreemds aan de hand. Omdat het aantal beeldpunten op de horizontale as gelijk is aan het aantal beeldpunten op de verticale as (192), zouden we een diagonaal doorgesneden vierkant mogen verwachten. Om het nog duidelijker te maken, typen we de volgende regels in:

```
10 SCREEN 2
20 LINE (0,0)-(100,100),2,BF
30 GOTO 30
```

Dit zou een vierkant op moeten leveren met zijden van 100 beeldpunten. Het vierkant is echter geen vierkant maar een rechthoek!
Herinnert u zich de cirkel nog die de toevoeging 1.35 nodig had om rond te worden? Dit verschijnsel doet zich hier ook voor (en ook bij gebruik van het lageresolutiescherm 3). Het is een nogal storende slordigheid in het MSX-systeem.
Er zijn twee mogelijkheden om dit euvel te verhelpen.

- Bij alle tekenopdrachten wordt de Y-coördinaat vermenigvuldigd met een factor 1.35.
- U verdraait de beeldhoogte-instelling van de TV (zit meestal achterop het toestel) tot het grafische scherm beeldvullend is. De onder- en bovenrand verdwijnen dan. Deze afregeling kan het beste gebeuren met de rechthoek uit het voorgaande programma in beeld. Dan is goed te beoordelen of de afregeling het gewenste resultaat heeft.

In dit boek is gekozen voor de eerste methode. Het is namelijk zeer onpraktisch om bij de wisseling van een tekstscherm naar een grafisch scherm de beeldhoogte opnieuw af te regelen. Als het toestel bovendien ook nog gebruikt wordt als TV-ontvanger, dan kunt u de grootste problemen met eventuele medegebruikers verwachten. Stel je voor, een weerman met een lang gezicht. Dat kan niet, ook al is er vaak een goede reden voor.
Een laatste variant van de LINE-opdracht is de stap-functie.

```
LINE(2Ø,7Ø)-STEP(3Ø,1Ø)
```

of:

```
LINE -STEP(3Ø,1Ø)
```

In het eerste geval wordt er een lijn getrokken vanaf beeldpunt (20,70) waarbij het eindpunt 30 beeldpunten naar rechts en 10 beeldpunten naar beneden komt te liggen. De tweede opdracht doet hetzelfde vanaf het beeldpunt dat het laatst door de computer is opgeroepen.
Hiermee hebben we de LINE-opdracht voldoende onder de loep genomen om er goed mee overweg te kunnen.
Ter afsluiting van dit hoofdstuk nog twee lijnenspelletjes.

```
10 REM verlichte stad
20 SCREEN 2:COLOR,1,1:CLS
30 PSET (120,60)
40 FOR I=1 TO 1000
50   X=RND(1)*10:Y=RND(1)*6:C=RND(1)*14
60   LINE -STEP(SGN(RND(1)-.5)*
     X,SGN(RND(1)-.5)*Y),C
70 NEXT I
80 RUN"CAS:"                    (80 GOTO 80)
```

Wat het voorstelt, is niet helemaal duidelijk: een landkaart, een asteroïdengordel, of een stad bij nacht? De laatste regel geeft alvast aansluiting met een volgende parel. In het allerlaatste pareltje van dit hoofdstuk lopen we alvast wat vooruit.

```
10  REM servetring
20  SCREEN 2:COLOR 10,4,10:CLS
30  FOR HOEK=.5*3.14 TO 1.5*3.14 STEP .1
40    GOSUB 100
50  NEXT HOEK
60  FOR HOEK=1.5*3.14 TO 2.5*3.14 STEP .03
70    GOSUB 100
80  NEXT HOEK
90  RUN"CAS:"                  (90 GOTO 90)
100 X=40*SIN(HOEK):Y=40*COS(HOEK)
110 PSET(130+X,90+Y)
120 LINE -STEP(0,50)
130 RETURN
```

Met behulp van een goniometrisch sommetje in regel 100 worden de punten van een cirkel berekend. Vanaf deze punten wordt een verticaal lijnstukje getrokken. Het programma laat zien dat het maken van berekeningen in BASIC erg traag gaat. In het volgende hoofdstuk worden de cirkels dan ook niet berekend. We gaan daar wat dieper in op het gebruik van de CIRCLE-opdracht, één van de mooiste grafische opdrachten die de MSX-computer kent.

9 Cirkels en aanverwante zaken

We zijn de cirkel-opdracht in dit boek al vele malen tegengekomen. In dit hoofdstuk blijven we er even bij stilstaan. De cirkel is een buitengewoon belangrijke vorm als we grafisch met de MSX-computer willen werken. Zoals we intussen gewend zijn, vallen we meteen met de deur in huis. We geven de computer geen tijd om af te koelen en typen in:

```
10 REM cirkels
20 SCREEN 2:COLOR 15,4,4:CLS
30 FOR 1 TO 100
40   CIRCLE(RND(1)*256,RND(1)*192),
     RND(1)*100
50 NEXT I
60 RUN"CAS:"              (60 GOTO 60)
```

We hadden al geconstateerd dat de op deze wijze verkregen cirkels niet rond zijn. Toch wel een mooi gezicht. Wellicht een goed idee voor behangpapier. Hoe de cirkels rond worden, is inmiddels ook duidelijk.

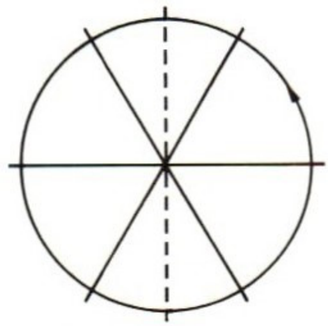
```
10 REM ronde cirkels
20 SCREEN 2:COLOR ,1,1:CLS
30 FOR 1 TO 100
40   CIRCLE(RND(1)*256,RND(1)*192),
     RND(1)*100,RND(1)*14,,,1.35
50 NEXT I
60 RUN"CAS:"             (60 GOTO 60)
```

In vergelijking met de CIRCLE-opdracht in het voorafgaande programma zijn er een paar toevoegingen. De beeldpuntcoördinaten tussen de haakjes en de straal direct achter de haakjes zijn onveranderd gebleven. Achter de straal is de gewenste kleur opgegeven. Vervolgens zijn er twee posities opengelaten. Op deze plaatsen kunnen de begin- en eindhoek worden ingevuld als we slechts een deel van de cirkel willen tekenen. Zorg ervoor geen enkele komma te vergeten. Als sluiter van de rij is het getal opgegeven dat de rondheid van de cirkel vastlegt. Dat wil zeggen de verhouding tussen de horizontale en de verticale as. Om op een normaal TV-beeldscherm een ronde cirkel te krijgen, moeten we, zoals we hebben gezien, hier 1.35 invullen.

Om het gebruik van de CIRCLE-opdracht ook voor de niet-wiskundige een beetje toegankelijk te maken, voeren we de appeltaart ten tonele.

Een cirkel wordt linksom getekend, waarbij als eerste punt het beeldpunt rechts van het middelpunt wordt getekend. Tenminste, als we er verder niets aan doen. Als een cirkel volledig getekend is, heeft de straal van de cirkel zich

over een hoek van 2*pi radialen bewogen. Als er slechts een deel van de cirkel hoeft te worden getekend, dan kan de programmeur met een wiskundige achtergrond de begin- en eindhoek opgeven in radialen. Door een beetje te smokkelen met de nauwkeurigheid van de hoeken kunnen we de wiskunde vergeten en de cirkel als een appeltaart beschouwen. Afbeelding 9-1 laat een royale zespersoons appeltaart zien.



Afbeelding 9-1 Een in zes stukken verdeelde cirkel.

We verdelen de taart in zes punten. De hoekpunten van de stukken nummeren we van 0 tot en met 6. Willen we de cirkelboog van taartpunt 0-1 tekenen, dan vullen we eenvoudig die twee getallen in. Zo dus:

```
CIRCLE(1ØØ,1ØØ),3Ø,6,Ø,1,1.35
       |   |    |  | | |  |
       x   y    r  | | |  rondheid
                   | | eindhoek
                   | beginhoek
                   kleur
```

De bovenste helft van een cirkel ziet er dan uit zoals in afbeelding 9-2.



Afbeelding 9-2

```
CIRCLE(1ØØ,1ØØ),3Ø,6,Ø,3,1.35
```

De cirkelboog loopt immers van 0 tot 3. De onderste helft van de cirkel staat in afbeelding 9-3.



Afbeelding 9-3

```
CIRCLE(1ØØ,1ØØ),3Ø,6,3,6,1.35
```

De boog loopt hier van 3 tot 6. Met even veel recht kunnen we zeggen dat de cirkelboog van 3 naar 0 loopt.

```
CIRCLE(1ØØ,1ØØ),3Ø,6,3,Ø,1.35
```

Het resultaat op het scherm is dan ook hetzelfde. Hiermee is dan meteen aangegeven hoe cirkeldelen in de rechterhelft worden opgegeven. We gaan gewoon door de 0 heen, waarbij we altijd linksom draaien. Van 5 tot 1 geeft dus een derde cirkelboog met de bolle kant naar rechts. In de notering mogen ook cijfers achter de komma (voor de computer is dat dus een punt) voorkomen. Dat is vaak handig als we ergens precies op aan willen sluiten.
Door gebruik te maken van de verdeling van 0 tot 6 maken we een kleine fout. Voor een volledige cirkel zou het laatste getal 6.28 moeten zijn. Dat is eenvoudig te controleren door twee cirkels te tekenen. Eén van 0 tot 6 en één van 0 tot 6.28.

```
10 REM cirkelbenadering
20 SCREEN 2:COLOR ,4,4:CLS
30 CIRCLE (70,90),50,15,0,6,1.35
40 CIRCLE (170,90),50,15,0,6.28,1.35
50 GOTO 50
```

Omdat we de begin- en eindhoek alleen maar opgeven als we cirkeldelen willen tekenen, is het gemaakte foutje bij de verdeling van 0 tot 6 zonder meer acceptabel. Zeker gezien het gemak dat het gebruik van ronde getallen met zich meebrengt. Het voorgaande programma is leerzaam, maar niet interessant genoeg om er een pareltje van te maken. Daarom rijgen we, voor we gaan oefenen met delen van cirkels, even een ander pareltje aan het snoer.

```
10 REM volle bollen
20 SCREEN 2:COLOR,4,4:CLS
30 FOR I=1 TO 30
40   X=RND(1)*256:Y=RND(1)*192:
     R=RND(1)*70:C=RND(1)*16
50   CIRCLE(X,Y),R,C,,,1.35
60   PAINT(X,Y),C
70 NEXT I
80 RUN"CAS:"          (80 GOTO 80)
```

We hebben aan onze ketting al een hele serie kleine pareltjes. Natuurlijke parels groeien in een oester. Ze zijn opgebouwd uit laagjes. Als we een parel langer laten groeien, kan er een prachtig groot exemplaar ontstaan. In het nu volgende deel van dit hoofdstuk gaan we laagsgewijs een grote parel opbouwen. Het wordt een juweeltje van een MSX-grafiek, voor een groot deel opgebouwd met de CIRCLE-opdracht.
Het is een mooie zomerse dag. De hemel is gevuld met kleine schapewolkjes.

```
10 REM schapewolkjes
20 SCREEN 2,2:COLOR 15,4,4:CLS
30 LINE(0,120)-(256,192),3,BF
40 FOR I=1 TO 150
50   X=RND(1)*256:Y=RND(1)*30:R=RND(1)*10
60   CIRCLE(X,Y),R,,3.9,5.5
70 NEXT I
80 GOTO 80
```

Dat ziet er al aardig uit. De cirkelbogen lopen van 3.9 tot 5.5 terwijl ze iets afgeplat zijn, omdat de toevoeging 1.35 weggelaten is. Ook de kleuraanduiding is weggelaten, zodat ze de inktkleur wit (15) uit regel 20 hebben. Het getal pi komt er niet aan te pas.
Zou het niet mooier zijn als enkele speelse elementen het nogal saaie landschap zouden opfleuren? Wat boompjes aan de horizon, een landweggetje, wat begroeiing hier en daar?
Zo'n uitbreiding laat zich, met de kennis die we nu van cirkels hebben, betrekkelijk eenvoudig aanbrengen.

```
10  REM landschap
20  SCREEN 2:COLOR 15,4,4
30  LINE(0,120)-(256,192),3,BF
40  FOR I=1 TO 150
50    X=RND(1)*256:Y=RND(1)*30:R=RND(1)*10
60    CIRCLE(X,Y),R,,3.9,5.5
70    CIRCLE(X,120),R,12,,3
80  NEXT I
90  COLOR 10
100 PSET(135,120)
130 DRAW"M160,192M100,192M135,120"
140 PAINT(135,125),10
150 FOR I=1 TO 10
160   CIRCLE(130-I*I/1.5,110+I*I/1.5),
      I*I/5,12,,,2
170   PAINT(130-I*I/1.5,110+I*I/1.5),12
180   LINE(130-I*I/1.5,110+I*I/1.5)-
      (131-I*I/1.5,110+I*I),12,BF
190   CIRCLE(140+I*I/1.5,110+I*I/1.5),
      I*I/5,12,,,2
200   PAINT(140+I*I/1.5,110+I*I/1.5),12
210   LINE(140+I*I/1.5,110+I*I/1.5)-
      (141+I*I/1.5,110+I*I),12,BF
220 NEXT I
230 GOTO 230
```

Laat u niet ontmoedigen door de schijnbaar ingewikkelde CIRCLE-,PAINT- en LINE-opdrachten. Ze lijken erg veel op elkaar en door het veranderen van het regelnummer en enkele details kunnen ze van elkaar worden gekopieerd. De hier gebruikte DRAW-opdracht is een hoofdstukje apart waard. Dat volgt dan ook nog. Als het programma zijn werk gedaan heeft, zult u verbaasd staan over de beheersing die we al over de grafische capaciteiten van de MSX-computer hebben. Wie zou er in zo'n vredig landschap niet een wandeling willen maken. Toch moet u iets opvallen. Het standpunt van waaruit u het landschap bekijkt, is nogal hoog. Zou het soms kunnen zijn dat u dit landschap vanuit een hoog flatgebouw bekijkt? Bovendien is er nog heel wat anders aan de hand!
Voeg de volgende regels maar eens toe:

```
222 FOR I=0 TO 256
224   PSET (I,60),14
226   FOR Q=1 TO 40:NEXT Q
228 NEXT I
```

Probeer voor het intypen van RUN eens te raden wat er zal gebeuren door deze toevoeging. Alleen het geluid ontbreekt nog.
Sporen van verkeersvliegtuigen worden gevormd door gecondenseerd, soms zelfs bevroren, water dat zich in het uitlaatgas bevindt (bij verbranding van kerosine komt water vrij). Vaak is, zoals in ons programma, het vliegtuig zelf niet eens te zien.
Toch zou het aardig zijn te kunnen beschikken over een echt vliegtuig. We gaan er een bouwen met behulp van een sprite. Hoe dat precies in zijn werk gaat, onderzoeken we in een volgend hoofdstuk. We typen, zonder er al te veel bij stil te staan, een aanvullend stukje programma. Voor we dat doen, verwijderen we de zojuist toegevoegde regels 222 tot en met 228.

```
10  REM landschap
20  SCREEN 2,2:COLOR 15,4,4
30  LINE(0,120)-(256,192),3,BF
40  FOR I=1 TO 150
50    X=RND(1)*256:Y=RND(1)*30:R=RND(1)*10
60    CIRCLE(X,Y),R,,3.9,5.5
70    CIRCLE(X,120),R,12,,3
80  NEXT I
90  COLOR 10
100 PSET(135,120)
130 DRAW"M160,192M100,192M135,120"
140 PAINT(135,125),10
150 FOR I=1 TO 10
160   CIRCLE(130-I*I/1.5,110+I*I/1.5),
      I*I/5,12,,,2
170   PAINT(130-I*I/1.5,110+I*I/1.5),12
180   LINE(130-I*I/1.5,110+I*I/1.5)-
      (131-I*I/1.5,110+I*I),12,BF
190   CIRCLE(140+I*I/1.5,110+I*I/1.5),
      I*I/5,12,,,2
```

```
200     PAINT(140+I*I/1.5,110+I*I/1.5),12
210     LINE(140+I*I/1.5,110+I*I/1.5)-
        (141+I*I/1.5,110+I*I),12,BF
220 NEXT I
```

Dit hadden we dus al. Let echter goed op. In regel 20 staat: SCREEN 2,2. We vertellen de computer hiermee dat we een grote sprite van 16 bij 16 beeldpunten in gaan voeren. De rest van het programma is nog niet eerder ingetypt.

```
230 DATA 0,0,14,133,194,239,112,64,254,194,
         132,9,14,0,0,0
240 DATA 0,0,0,0,128,252,2,1,127,64,128,0,
         0,0,0,0
250 FOR S=1 TO 32
260   READ A
270   S$=S$+CHR$(A)
280 NEXT S
290 SPRITE$(0)=S$
300 SOUND 7,55
310 SOUND 8,16
320 SOUND 11,255:SOUND 12,120:SOUND 13,13
330 FOR I=0 TO 255
340   SOUND 6,INT(I/9)
350   IF I=170 THEN SOUND 13,0
360   PUT SPRITE 0,(I,52),15,0
370   PSET(I-10,60),15
380   PSET(I-50,60),14
390   PSET(I-90,60),4
400 NEXT I
410 RUN"CAS:"                  (410 GOTO 410)
```

Dit ruw verstoord landelijk tafereeltje is voor een parel nogal fors. Toch is het verwonderlijk dat we in minder dan veertig regeltjes bijzonder mooie achtergronden kunnen maken waarin we elementen kunnen laten bewegen. Enkele schapen in de wei, een vlindertje in de lucht en meer van dat moois wordt pas echt goed mogelijk als we het ontwerpen van en manipuleren met sprites onder de knie hebben. Daar besteden we dan ook een uitgebreid hoofdstuk aan.
Om de (rijg)draad van dit boek weer op te pakken, spelen we nog even met CIRCLE, zonder een andere bedoeling dan zomaar wat aardige prentjes te maken.
We hebben al eens eerder een logo van een groot bedrijf gemaakt op het lageresolutiescherm. In hoog oplossend vermogen (SCREEN2), ziet dat er zó uit:

```
10 REM logo
20 SCREEN 2:COLOR 10,4,4:CLS
30 OPEN"GRP:" FOR OUTPUT AS #1
40 PSET(103,87),4
50 PRINT#1,"DuPont"
60 CIRCLE(124,90),30,,,,.5
70 RUN"CAS:"          (70 GOTO 70)
```

Let weer goed op de punten en de komma's.
Kent u het volgende verkeersbord?

```
10  REM verkeer
20  SCREEN 2:COLOR ,1,1:CLS
30  OPEN"GRP:" FOR OUTPUT AS #1
40  CIRCLE(130,100),90,6,,,1.35
50  PAINT(130,100),6
60  CIRCLE(130,100),70,15,,,1.35
70  PAINT(130,100),15
80  PSET(0,0),1
90  PRINT#1,"VERKEER"
100 RUN"CAS:"      (100 GOTO 100)
```

Nu draaien we alle kleurnummers om.

```
10  REM soep
20  SCREEN 2:COLOR ,1,1:CLS
30  OPEN"GRP:" FOR OUTPUT AS #1
40  CIRCLE(130,100),90,15,,,1.35
50  PAINT(130,100),15
60  CIRCLE(130,100),70,6,,,1.35
70  PAINT(130,100),6
80  PSET(0,0),1
90  PRINT#1,"TOMATENSOEP"
100 RUN"CAS:"      (100 GOTO 100)
```

Een oud grapje, dat wel. Maar toch mooi geïllustreerd. Kent u het volgende bord?

```
10  REM verkeer
20  SCREEN 2:COLOR ,15,15:CLS
30  OPEN"GRP:" FOR OUTPUT AS #1
40  CIRCLE(130,100),90,6,,,1.35
60  CIRCLE(130,100),70,6,,,1.35
70  PAINT(130,171),6
80  LINE(80,125)-(180,45),6
90  LINE(85,145)-(185,65),6
100 PAINT(128,96),6
110 RUN"CAS:"      (110 GOTO 110)
```

In deze miniprogramma's komt de PAINT-opdracht veelvuldig voor. We zullen er op de nu volgende pagina's een hoofdstukje aan wagen.

10 De schilderskwast ter hand genomen

Met de PAINT-opdracht kunnen we vlakken schilderen. Die vlakken mogen alle vormen aannemen. Er zijn echter drie voorwaarden waaraan moet worden voldaan:

- De contouren die het in te kleuren vlak bepalen, moeten volledig gesloten zijn.
- De lijn waarmee de contour is getekend, moet de kleur hebben waarmee wordt ingekleurd.
- De coördinaten in de PAINT-opdracht moeten binnen het vlak vallen en mogen niet op de buitenste rand van beeldpunten vallen.

Laten we met een voorbeeld beginnen:

```
10 REM glas
20 SCREEN 2:COLOR 15,15,15:CLS
30 LINE(0,130)-(256,192),12,BF
40 CIRCLE(130,40),35,6,,,.3
50 CIRCLE(130,170),26,6,,,.3
60 CIRCLE(130,100),31,6,,,.3
70 LINE(94,40)-(104,170),6
80 LINE(165,40)-(156,170),6
90 GOTO 90
```

De bovenstaande regels zorgen ervoor dat er een realistisch getekend glas op tafel komt te staan. Stel dat we dit glas willen vullen met tomatensap, dan moeten we drie PAINT-opdrachten geven omdat er drie gesloten contouren aanwezig zijn die moeten worden gevuld.

```
52 PAINT(130,170),6
62 PAINT(130,100),6
82 PAINT(130,135),6
```

Het kan ook anders. Als we een gaatje prikken in de twee ellipsen die de begrenzing van de vloeistof aangeven, dan kunnen we met slechts één PAINT-opdracht volstaan. De verf lekt dan door die gaatjes vanzelf in de andere delen van de tekening. Dat doorprikken van een ellips gaat het best met een naald in de achtergrondkleur. Omdat er twee achtergrondkleuren zijn, hebben we twee naaldjes nodig. We halen de regels 52, 62 en 82 weer weg en voegen de naalden toe.

```
82 LINE(130,100)-(130,130),15
84 LINE(130,130)-(130,170),12
86 PAINT(130,177),6
88 CIRCLE(130,100),31,9,,,.3
```

Het glas wordt nu gevuld door één enkele PAINT-opdracht. Dat laatste lichtrode randje maakt er een smakelijk uitziend drankje van. De uitgelopen groepjes van acht beeldpunten geven er zelfs iets levendigs aan. Na RENUM en de toevoeging van de koppeling met een volgende parel ziet de 'Bloody Mary' er zó uit:

```
10  REM glas
20  SCREEN 2:COLOR 15,15,15:CLS
30  LINE(0,130)-(256,192),12,BF
40  CIRCLE(130,40),35,6,,,.3
50  CIRCLE(130,170),26,6,,,.3
60  CIRCLE(130,100),31,6,,,.3
70  LINE(94,40)-(104,170),6
80  LINE(165,40)-(156,170),6
90  LINE(130,100)-(130,130),15
100 LINE(130,130)-(130,170),12
110 PAINT(130,177),6
120 CIRCLE(130,100),31,9,,,.3
130 OPEN"GRP:" FOR OUTPUT AS #1
140 PSET(0,10),15
150 COLOR 1
160 PRINT#1,"GEZONDHEID"
170 RUN"CAS:"     (170 GOTO 170)
```

Voor de afwisseling rijgen we een fantasieprogramma aan de draad, waarin PAINT een hoofdrol speelt.

```
10  REM papier
20  SCREEN 2:COLOR ,4,4:CLS
30  FOR I=1 TO 5
40    COLOR I+8
50    PSET(100,170-30*I)
60    LINE -STEP(-50,50)
70    LINE -STEP(100,0)
80    LINE -STEP(50,-50)
90    LINE -STEP(-100,0)
100   PAINT(110,180-30*I)
110 NEXT I
120 RUN"CAS:" (120 GOTO 120)
```

Dit programma demonstreert weer eens de ongekende mogelijkheden van de MSX-computer op het gebied van grafische weergave. Let eens op het nuanceverschil tussen geel en lichtgeel. Verander nu de kleurnummers 4 in regel 20 in 1. De achtergrond is nu zwart in plaats van blauw. Het nuanceverschil tussen geel en lichtgeel is nu grotendeels verdwenen. Bij gebruik van een duurdere monitor treden dit soort kleurverschuivingen niet op. Als homecomputerhobbyisten zullen we er mee moeten leren leven.
Met kleine veranderingen in het bovenstaande programma zijn prachtige resultaten te behalen.

```
10  REM nog meer papier
20  SCREEN 2:COLOR ,4,4:CLS
30  FOR I=2 TO 15
40    COLOR I
50    PSET(100,150-10*I)
60    LINE -STEP(-50,50)
70    LINE -STEP(100,0)
80    LINE -STEP(50,-50)
90    LINE -STEP(-100,0)
100   PAINT(110,180-10*I)
110 NEXT I
120 RUN"CAS:" (120 GOTO 120)
```

Een pak vouwblaadjes voor de kleuterschool.
In het lageresolutiescherm (3) kan aan PAINT nog een opdracht worden toegevoegd. Behalve de X- en Y-coördinaat en de verfkleur, kan ook de kleur van de omtrek worden opgegeven. Heeft u daar behoefte aan?
We maken op de PAINT-valreep gauw nog een pareltje voor de ketting. De stapel papier wordt op kleurvolgorde gelegd.

```
10  REM nog veel meer papier
20  SCREEN 2:COLOR ,1,1:CLS
30  FOR I=1 TO 14
40    READ C:COLOR C
50    PSET(100,150-10*I)
60    LINE -STEP(-50,50)
70    LINE -STEP(100,0)
80    LINE -STEP(50,-50)
90    LINE -STEP(-100,0)
100   PAINT(110,180-10*I)
110 NEXT I
120 DATA 12,2,3,9,8,6,13,4,5,7,14,15,10,11
130 RUN"CAS:"                  (130 GOTO 130)
```

Ten opzichte van de vorige stapel zijn de regels 30 en 40 aan verandering onderhevig geweest. Bovendien zijn de kleurnummers in een DATA-regel opgenomen. Het kleurenpalet van de MSX-computer is heel bruikbaar, al laat het hier en daar toch wel te wensen over. Zo ontbreekt de kleur oranje. In het volgende hoofdstuk proberen we zelf oranje en andere mengkleuren te maken.

11 Een cursus verfmengen

Om de kleur oranje te maken, moeten we op de een of andere wijze de kleuren rood en geel in een bakje doen en flink roeren.
Op het beeldscherm is de beeldpunt het kleinste element. We kunnen proberen oranje te maken door de beeldpunten om en om rood en geel te maken. De mengkleur van rood en geel is immers oranje. Het volgende programma laat zien hoe dat werkt.

```
10 REM oranje
20 SCREEN 2:COLOR ,6,6:CLS
30 FOR Y= 50 TO 100
40   FOR X=0 TO 256 STEP 2
50     PSET(X,Y),10
60 NEXT X,Y
70 GOTO 70
```

Dat werkt dus niet. Natuurlijk niet, zult u denken.
Door de eerder behandelde horizontale groepjes van acht beeldpunten moeten we als achtergrondkleur één van de te mengen kleuren nemen. Er kunnen dan, per groepje van acht horizontale beeldpunten, nooit meer dan twee kleuren voorkomen.

```
10 REM oranje?
20 SCREEN 2:COLOR ,4,4:CLS
30 FOR Y=50 TO 100
40   FOR X=0 TO 256 STEP 2
50     PSET(X,Y),10:PSET(X,Y),6
60 NEXT X,Y
70 GOTO 10
```

Het kan dus toch! Oranje op de buis. Ondanks dat de opbouw van de kleur nogal traag gaat, kunnen we de kleur oranje die we zojuist hebben gemaakt toepassen in programma's.
Een klein programma bewijst weer eens dat er meer in de MSX-computer zit dan er in de handboeken staat.

```
10  REM hoera
20  SCREEN 2:COLOR ,6,3:CLS
30  FOR Y=40 TO 60
40    FOR X=48 TO 150 STEP 2
50      PSET(X,Y),10
60  NEXT X,Y
70  LINE(48,70)-(150,90),6,BF
80  LINE(48,90)-(150,110),15,BF
```

```
90  LINE(48,110)-(150,130),4,BF
100 LINE(47,39)-(152,61),3,B
110 LINE(47,69)-(151,131),3,B
120 PAINT(1,1),3
130 LINE(44,30)-(47,180),1,B
140 RUN"CAS:"    (140 GOTO 140)
```

Door een truc uit te halen, kunnen we dus toch mengkleuren maken tegen elke gewenste achtergrondkleur, die zelf ook weer een mengkleur kan zijn. De mengkleur wordt afgeschermd met een lijntje in de gewenste achtergrondkleur (de regels 100 en 110) en een PAINT-opdracht doet de rest. De coördinaten van de PAINT-opdracht moeten dan natuurlijk buiten de reeds ingekleurde rechthoeken liggen. Om onze artistieke talenten op de computer uit te kunnen leven, zullen we eens wat kleurtjes mengen. Uit een stuk triplex zagen we een palet, waarop we de te mengen kleuren uitspreiden. Door het palet in de kleur te schilderen waarmee we enkele andere kleuren willen mengen, kan het programma relatief eenvoudig blijven. Penseel bij de hand?

```
10  REM palet
20  SCREEN 2:COLOR 7,7,1:CLS
30  FOR Y=10 TO 70
40    FOR X=16 TO 88 STEP 2
50      PSET(X,Y),4
60  NEXT X,Y
70  FOR Y=45 TO 105
80    FOR X=103 TO 175 STEP 2
90      PSET(X,Y),9
100 NEXT X,Y
110 FOR Y=130 TO 190
120   FOR X=160 TO 232 STEP 2
130     PSET(X,Y),3
140 NEXT X,Y
150 CIRCLE(50,192),200,1
160 PAINT(255,1),1
170 PSET(0,187)
180 LINE -STEP(70,-40),1
190 LINE -STEP(3,10),1
200 LINE -STEP(-50,34),1
210 PAINT(1,191),1
220 PSET(70,147),1
230 DRAW"U3E2U2E2U3E4U3E5R12F4R5F6"
240 DRAW"L6G9D2G4D3G3G1L6D1L6"
250 RUN"CAS:"        (250 GOTO 250)
```

Dat begint er op te lijken. Toch doen de vierkante verfvoorraadjes wat vreemd aan. Niet gek voor een kubist, maar Rembrandt zou er moeite mee hebben gehad. Bovendien kan er alleen maar worden gemengd met een enkele grondkleur. Nu we de basisopzet voor dit programma eenmaal hebben, is het niet moeilijk meer de scherpe kantjes er af te halen. Let goed op details zoals coördinaten en kleurnummers.

```
10  REM palet 2
20  SCREEN 2:COLOR 15,4,4:CLS
30  PSET(16,10):LINE -STEP(72,60),6,BF
40  PSET(103,45):LINE -STEP(72,60),13,BF
50  PSET(160,128):LINE -STEP(72,60),3,BF
60  FOR Y=10 TO 70
70    FOR X=16 TO 88 STEP 2
80      PSET(X,Y),10
90  NEXT X,Y
100 FOR Y=45 TO 105
110   FOR X=103 TO 175 STEP 2
120     PSET(X,Y),7
130 NEXT X,Y
140 FOR Y=130 TO 190
150   FOR X=160 TO 232 STEP 2
160     PSET(X,Y),15
170 NEXT X,Y
180 CIRCLE(50,40),30
190 CIRCLE(142,75),30
200 CIRCLE(195,159),30
210 CIRCLE(50,192),192
220 PAINT(1,191)
230 PSET(0,187)
240 LINE -STEP(70,-40),1
250 LINE -STEP(3,10),1
260 LINE -STEP(-50,34),1
270 PAINT(1,191),1
280 PSET(70,147),1
290 DRAW"U3E2U2E2U3E4U2E5R12F4R5F6"
300 DRAW"L6G9D2G4D3G3G1L6D1L6"
310 RUN"CAS:"          (310 GOTO 310)
```

De meeste scherpe kantjes zijn nu van de verfvlekken af. Toch speelt het regeltje van acht ons weer parten. Omdat we niet kunnen vermijden dat er meer dan twee kleuren per acht horizontale beeldpunten voorkomen, zullen we met deze afwijking van het volmaakte moeten leren leven. Toch is het onverwacht dat we met de MSX-computer zelfs pasteltinten op het scherm kunnen toveren.

In het volgende hoofdstuk kijken we naar een opdracht waarvoor in MSX-BASIC een op zichzelf staand minitaaltje is ontwikkeld. Het is de DRAW-opdracht, een combinatie van de mogelijkheden van PSET, LINE en COLOR. We zijn deze opdracht al een paar keer tegengekomen.

12 Roept u maar!

De DRAW-opdracht luistert naar woorden. Het is een soort robot die zich op commando over het scherm beweegt. We roepen: "drie plaatsen naar links", en DRAW voert de opdracht die tussen aanhalingstekens staat keurig uit. In BASIC worden karakterreeksen strings genoemd; deze staan tussen aanhalingstekens. Een variabele die een string representeert, is te herkennen aan het dollar-teken ($) achter zijn naam. De uitroep "drie plaatsen naar links" is dus een string.
Speciaal voor de DRAW-opdracht is een taaltje ontwikkeld, een soort vakjargon, dat ons in staat stelt de robot naar elke plaats op het scherm te bewegen. Op die manier kunnen we een tekening maken.

```
10 REM DRAW
20 SCREEN 2:COLOR 15,4,4:CLS
30 PSET(100,50)
40 DRAW"R70D40L60U30R50D20L40U10R20"
50 GOTO 50
```

In regel 40 is te zien dat we met een letter de richting en met een getal het aantal beeldpunten op kunnen geven. Zo betekent 'L40' ga veertig beeldpunten naar links. Omdat BASIC een internationaal gebruikte taal is, worden de richtingen in het Engels aangeduid.

Omhoog	Up	U
Omlaag	Down	D
Rechts	Right	R
Links	Left	L

Diagonaal bewegen kan ook. De ontwerper van de DRAW-subtaal is echter niet in staat geweest hier gemakkelijk te onthouden benamingen voor te verzinnen.

Rechts	Omhoog	E
Links	Omhoog	H
Rechts	Omlaag	F
Links	Omlaag	G

Behalve de één-assige verplaatsingsopdrachten kunnen ook coördinaten worden opgegeven. In dat geval gebruiken we de MOVE-opdracht.

```
M 1Ø,3Ø
```

trekt een lijn naar het punt met de coördinaten (10,30). Evenals bij de LINE-opdracht kan in MOVE ook een STEP-functie worden opgenomen. Bij MOVE kan dit ook afzonderlijk voor beide coördinaten.

```
M-1Ø,3Ø
```

trekt een lijntje 10 beeldpunten naar links en naar Y-coördinaat 30.

```
M1Ø,+3Ø
```

betekent een lijn naar X-coördinaat 10 en een verplaatsing van 30 beeldpunten omlaag. Zodra we een verplaatsing nodig hebben zonder dat er een lijn wordt getrokken, brengt de letter B uitkomst.

```
10 SCREEN 2
20 PSET(100,100)
30 DRAW"BM-50,50"
40 DRAW"C10R100"
50 GOTO 50
```

Kijk nog even naar regel 40. Daar staat een nieuw type opdracht. Met C kunnen we een kleur aangeven. In dit geval geel (10).
Nog een paar mogelijkheden:

```
10 SCREEN 2:COLOR 15,4,4:CLS
20 A$="R8U6L8D6"
30 B$="L8D6R8U6"
40 DRAW"BM130,90"
50 FOR I=1 TO 50
60   DRAW"S=I;XA$;XB$;"
70 NEXT I
80 RUN"CAS:"          (GOTO 80)
```

Een aardig effect. In de regels 20 en 30 worden de bewegingsopdrachten buiten DRAW alvast opgebouwd. Uiteraard in de vorm van een string. Regel 40 brengt de tekenpunt naar het midden van het scherm. We zouden hier evengoed een PSET-opdracht voor kunnen gebruiken. In regel 60 wordt de opdracht uitgevoerd. S geeft de schaalfactor aan waarmee de getallen in de bewegingsopdrachten worden vermenigvuldigd. Omdat I toeneemt, ontstaan er steeds grotere rechthoeken. Als we gegevens buiten de DRAW-opdracht naar binnen willen halen, kan dat door er een X voor te zetten. In feite kan elke DRAW-opdracht ook met LINE en PSET worden uitgevoerd. Met DRAW echter kan compacter worden geprogrammeerd.
Regel 90 maakt dit werkje tot een pareltje voor het al indrukwekkend lang geworden rijgsnoer. Neem af en toe de moeite het complete oeuvre eens terug te spelen. U zult getroffen worden door de variëteit aan beelden die al op het bandje staan. Dat is nog eens wat anders dan vakantiedia's. Schrik er niet voor terug ook anderen lastig te vallen met uw werkstuk. Het is beter van kwaliteit

dan menige TV-serie. Zelf ontworpen STER-spotjes doen het trouwens ook erg goed aan een parelketting.
Vanwege de beperking tot rechte lijnen zullen we verder niet stilstaan bij de DRAW-opdracht. Dit boek is niet bedoeld om alle onderwerpen uitputtend te behandelen. Tot nu toe hebben we verschillende soorten lijnen getekend, recht en geknikt, cirkelvormen en ellipsen. Toch is er een type lijn dat nog niet geheel uit de verf is gekomen: tekenlijn die willekeurige vormen kan volgen. Het palet uit het vorige hoofdstuk liet een penseel zien. De kwast ervan werd gevormd door zo'n grillig gevormde lijn. In dit geval werd die lijn opgebouwd met de DRAW-opdracht, hetgeen toch tamelijk moeilijk gaat.
In het volgende hoofdstuk bekijken we een methode om willekeurige tekeningen te produceren. Prentjes bijvoorbeeld, die al op papier staan en waarvan we graag een kopie op het scherm willen hebben. Afbeeldingen van familieleden, een favoriete stripfiguur. Of gewoon de plattegrond van uw huis als basis voor een spannend avonturenspel.

13 Vrij tekenen

In alle voorafgaande hoofdstukken hebben we gebruik gemaakt van voorgeprogrammeerde MSX-functies. We hebben gezien dat er heel wat grafisch spektakel mee kan worden opgewekt. We zijn echter ook beperkingen tegengekomen. Zo liet het vorige hoofdstuk zien dat de DRAW-opdracht zijn naam niet echt waarmaakt. Erg ingewikkelde tekeningen zijn er niet mee te maken. In dit hoofdstuk gaan we een gereedschap ontwikkelen waarmee we elke tekening kunnen maken die uit lijnen is opgebouwd. Om de nieuwsgierigheid te prikkelen, volgt een programma om een tekening te maken. De tekening wordt gemaakt volgens de methode die in dit hoofdstuk staat beschreven. Schrik niet van de enorme lijst getallen in de DATA-regels. Het zijn, op enkele uitzonderingen na, paarsgewijs alle X- en Y-coördinaten waaruit de tekening bestaat. Normaal gesproken wordt deze getallenrij automatisch op een bandje gezet en er vanzelf weer afgelezen.
Het in dit hoofdstuk te ontwikkelen tekenprogramma is dan ook zeer gebruikersvriendelijk. Fouten kunnen worden hersteld en de resultaten kunnen erg aantrekkelijk zijn. Dat bewijst bijvoorbeeld de tekening die op het scherm verschijnt bij het uitvoeren van het volgende programma. Toegegeven, het plaatje is nogal pikant. Misschien wel te gewaagd voor een computerboek. Maar niemand dwingt u de indrukwekkende getallenrij in te voeren. Het risico is geheel aan u. Slimmerikken zullen wellicht voorzichtig proberen slechts enkele regeltjes data in te typen, om op die manier een idee te krijgen wat voor vlees we in de kuip hebben. Wees gewaarschuwd, u bent verloren!

```
10  REM Lorelei
20  SCREEN 2:COLOR 1,10,10:CLS
30  DIM X(200):DIM Y(200)
40  FOR D=1 TO 14
50    READ Q
60    FOR I=1 TO Q
70      READ X(I),Y(I)
80      IF I<2 THEN 100
90      LINE(X(I-1),Y(I-1))-(X(I),Y(I))
100   NEXT I
110 NEXT D
120 RUN"CAS:"                    (120 GOTO 120)

150 DATA 28,111,149,116,137,120,124,123,110,
         125,105,125,105,120,99,125,105,125,
         105,128,89,132,75,136,63
160 DATA 139,55,139,55,142,49,146,44,152,41,
         158,42,162,45,165,51,167,59,169,73,
         172,91,177,116,182,139,188,157,195,
         174,199,184
```

```
170 DATA 19,159,42,163,37,166,34,173,32,177,
         34,182,42,184,51,188,75,188,75,191,
         108,191,108,192,133,192,133,192,133,
         194,157,196,164,200,174,201,180,204,
         185
180 DATA 19,147,64,148,67,147,73,147,73,147,
         79,148,91,150,102,153,113,155,118,
         162,135,169,150,174,163,174,169,174,
         169,173,174,173,182,173,182
190 DATA 176,189,176,189,99,154,117,154,130,
         153,147,152,157,151,162,150,163,150,
         169,150,173,149,177,146,183,141,186,
         133,186,121,183,117,180,117,180,111
200 DATA 174,108,170,104,161,100,153,94,146,
         88,140,80,133,71,122,65,112,69,102,
         66,111,63,118,63,118,63,124,62,132,
         54,134,61,132,68,137,71,138
210 DATA 75,138,81,138,87,142,93,151,102,172,
         102,172,102,174,101,174,94,160,100,
         175,99,177,97,176,91,162,96,179,94,
         178,93,177,87,163,90,177,89
220 DATA 178,87,175,84,167,84,164,81,161,79,
         155,77,152,77,152,69,153,59,152,59,
         152,52,149,48,144,48,163,48,126,48,
         126,49,119,49,114,49,114,50,103,51
230 DATA 93,51,93,53,85,53,85,57,77,62,72,67,
         70,67,70,72,62,72,62,74,64,70,51,70,
         51,70,48,70,48,66,34,68,28,73,21,74,
         19,75,19,75,19,75,14,82,16,75,6
240 DATA 75,6,68,4,66,7,21,68,9,62,5,62,5,58,
         7,51,14,51,14,46,54,46,24,46,33,46,
         37,48,40,48,47,52,60,57,78,59,72,62,
         69,64,67,64
250 DATA 65,64,65,63,61,63,61,11,169,150,168,
         157,165,165,161,173,157,176,151,177,
         150,179,151,169,153,163,161,153,159
260 DATA 13,79,16,80,18,81,19,81,20,80,21,79,
         22,77,21,82,24,83,25,83,26,82,28,81,
         29,81,29
270 DATA 61,81,21,84,24,85,26,86,30,86,32,84,
         32,83,33,82,35,82,36,81,36,84,37,85,
         36,86,34,86,33,89,37,89,39,89,41,89,
         41,86,45,84,46,83,47,79,48,75
280 DATA 48,84,46,84,46,84,52,84,55,85,58,89,
         60,95,60,95,60,10,61,105,63,107,66,
         107,66,109,72,109,74,106,74,109,74,
         112,76
290 DATA 114,76,119,79,120,80,120,80,120,81,
         122,80,122,80,121,82,122,84,123,87,
         123,91,122,95,120,97,120,98,124,104,
         119,98,113,98,113,98,110
300 DATA 97,110,97,107,95
310 DATA 4,71,31,73,30,73,30,74,28
320 DATA 4,82,68,85,72,85,72,87,71
```

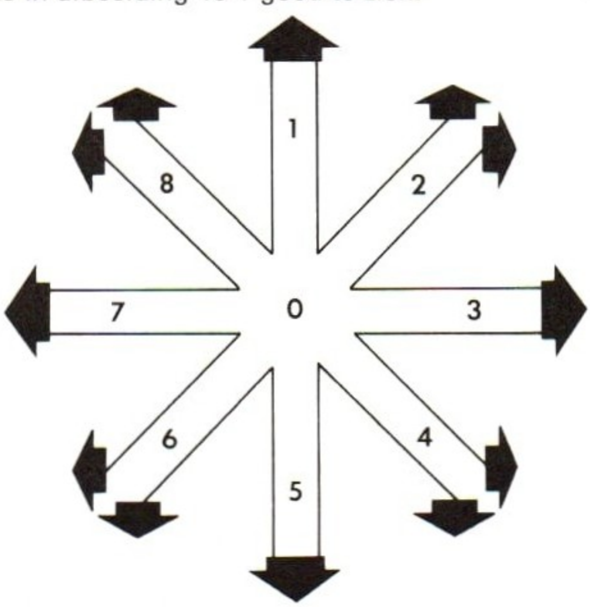
```
330 DATA 6,83,101,84,104,86,105,88,105,89,
         102,89,100
340 DATA 3,118,81,118,83,120,85
350 DATA 21,75,95,74,100,74,107,76,112,76,
         112,80,115,80,115,83,116,83,116,87,
         116,87,116,92,113,95,109,95,109,98,
         104,99,100,99,100,99,96,99,96,98,92,
         98,92
360 DATA 2,86,100,86,101
```

Let in dit geval extra op regel 120. Tijdens het proberen kan er beter 120 GOTO 120 staan, waardoor wordt voorkomen dat Lorelei verloren gaat. Het prentje komt overigens zeer goed in aanmerking voor een plaats in de illustere rij van parels die we al geregen hebben. De keuze of het ook werkelijk deel gaat uitmaken van de verzameling, is aan u.

Terug naar de werkelijkheid. Hoe krijgen we de computer zó ver dat hij ons in staat stelt een beeldpunt op het beeldscherm te manipuleren? Elke MSX-computer is uitgerust met vier cursorbesturingstoetsen (pijltoetsen). De MSX-computer behandelt deze toetsen en een joystick op dezelfde manier. De uitleesopdracht voor de cursorbesturingstoetsen is:

```
STICK(Ø)
```

STICK(1) en STICK(2) kijken naar de stand van de joysticks. Als u de joysticks prefereert boven de cursorbesturingstoetsen, dan is daar natuurlijk geen enkel bezwaar tegen, maar dan moeten wel de in de volgende programma's voorkomende STICK-opdrachten van het goede nummer worden voorzien. De STICK(0)-opdracht kijkt dus welke pijltoetsen zijn ingedrukt. Op welke manier dat gaat, is in afbeelding 13-1 goed te zien.



Afbeelding 13-1 De richtingen plus bijbehorende cursor-toetsen.

Elke richting komt overeen met een nummer. Een kort programma illustreert het gebruik.

```
10 CLS:KEYOFF
20 A=STICK(0)
30 LOCATE 15,10:PRINT A
40 GOTO 20
```

Met wat extra regels kan er al een figuurtje over het scherm worden bewogen.

```
10 CLS:KEYOFF
20 A=STICK(0)
30 IF A=3 THEN X=X+1
40 LOCATE X,10
50 PRINT CHR$(249)
60 GOTO 20
```

Regel 30 kan voor alle richtingen worden herhaald. Het is echter aantrekkelijker om een en ander te combineren.

```
10 SCREEN 1:CLS:KEYOFF
20 A=STICK(0)
30 IF A=2 OR A=3 OR A=4 THEN X=X+1
40 IF A=6 OR A=7 OR A=8 THEN X=X-1
50 IF A=8 OR A=1 OR A=2 THEN Y=Y-1
60 IF A=4 OR A=5 OR A=6 THEN Y=Y+1
70 LOCATE X,Y
80 PRINT CHR$(249)
90 GOTO 20
```

De werking van de vuurknop missen we nog. Voeg daarom een regel 85 aan het programma toe:

```
IF STRIG(Ø) THEN PRINT CHR$(2Ø4)
```

De spatiebalk heeft hierbij de functie van vuurknop. Als de spatiebalk ingedrukt is, levert STRIG(0) het getal −1 op. In de binaire algebra heet dat negatieve logica. Als de uitdrukking waar is (de knop is ingedrukt), dan is de waarde van die uitdrukking −1.
Op deze manier kunnen we ook met getallen werken. Als A=3 dan is (A=3)=−1, of met woorden: Als A=3 waar is, dan heeft de uitdrukking (A=3) de waarde −1.
Met deze kennis van negatieve logica is het vorige programma op elegante wijze te herschrijven.

```
10  SCREEN 1:CLS:KEYOFF
20  A=STICK(0)
30  X=X+(A=6)+(A=7)+(A=8)-(A=2)-(A=3)-(A=4)
40  Y=Y+(A=2)+(A=1)+(A=8)-(A=4)-(A=5)-(A=6)
```

```
50  IF Y>22 THEN Y=22
60  IF Y<0 THEN Y=0
70  IF X<0 THEN X=0
80  IF X>28 THEN X=28
90  LOCATE X,Y
100 PRINT CHR$(249);
110 IF STRIG(0)=-1 THEN PRINT "VUUR!"
120 GOTO 20
```

We hebben nu voldoende kennis verzameld om het tekenprogramma te kunnen volgen. Dit relatief kleine programma is in staat om een gigantische hoeveelheid coördinaten van beeldpunten te genereren en op cassetteband op te slaan, zonder dat we daar verder zelf iets aan hoeven te doen.

```
10  REM vrij tekenen
20  STOP ON
30  DIM X(200):DIM Y(200)
40  SCREEN 2:COLOR 15,4,4:CLS
50  XN=0:YN=0:XO=0:YO=0
60  X=0:Y=0
70  PSET (XN,YN)
80  X=XN:Y=YN
90  A=STICK(0)
100 XN=X+(A=6)+(A=7)+(A=8)-(A=2)-(A=3)-(A=4)
110 YN=Y+(A=2)+(A=1)+(A=8)-(A=4)-(A=5)-(A=6)
120 PRESET(X,Y):PSET(XN,YN)
130 ON STOP GOSUB 220
140 IF STRIG(0)=-1 THEN 150 ELSE 80
150 X(I)=XN:Y(I)=YN
160 I=I+1
170 IF I=200 THEN 170
180 LINE(XO,YO)-(XN,YN)
190 IF XO=0 AND YO=0 THEN LINE (XO,YO)-(XN,YN),4
200 XO=XN:YO=YN
210 GOTO 80
220 SCREEN 0:CLS
230 Q=I
240 FOR I=0 TO Q-1
250   PRINT X(I);Y(I);
260 NEXT I
270 PRINT:PRINT"OPSLAAN ? (J/N)";
280 B$=INPUT$(1)
290 IF B$="J" OR B$="j" THEN 300 ELSE 350
300 OPEN"CAS:"FOR OUTPUT AS #1
310 PRINT#1,Q
320 FOR I=0 TO Q-1
330   PRINT#1,X(I),Y(I)
340 NEXT I
350 CLOSE
360 SCREEN 0:PRINT"Nieuwe lijn ? (J/N)";
370 A$=INPUT$(1)
380 IF A$="J" OR A$="j" THEN RUN
390 IF B$="J" OR B$="j" THEN 400 ELSE END
```

```
400 ST=300
410 OPEN"CAS:" FOR OUTPUT AS #1
420 PRINT#1,ST
430 CLOSE
440 END
```

Het programma is zoals gezegd eenvoudig in het gebruik. Behalve de computer, geladen met het programma, hebben we nog enkele andere attributen nodig. Om te beginnen de afbeelding die we op het scherm willen brengen. Het effect is het mooist als de voorstelling ongeveer dezelfde afmetingen heeft als het beeldscherm. We kunnen de tekening dan op ware grootte op het beeldscherm overnemen. De tekening trekken we met een viltstift over op een stuk helder huishoudfolie. Elk helder folie is goed. Het folie met de overgetrokken lijntekening wordt met een plakbandje op de buis geplakt. Dat gaat het beste als de schermrand een afwijkende kleur heeft.

```
SCREEN 2:COLOR,4,5:CLS
```

Nu kan de juiste plaats van de afbeelding beter worden bepaald.
Het tekenprogramma kan nu worden gestart met RUN (F5). Het scherm wordt blauw en linksboven staat een witte punt. Deze punt kan met de pijltoetsen over het gehele scherm worden bewogen. We brengen de punt op de plaats waar een lijn begint en drukken dan op de spatiebalk. Vervolgens wordt de punt een stukje verderop op de lijn gezet en de spatiebalk wordt weer ingedrukt. Daar is het eerste lijnstukje. We werken zo de hele lijn af. Op rechte stukken verbinden we punten ver uit elkaar, op sterk gekromde lijnen leggen we de punten heel dicht bij elkaar. De 'tekenpunt' kan vrijelijk over het scherm worden bewogen tot u er zeker van bent dat hij op de juiste plaats staat. Druk dan pas op de spatiebalk. Is de lijn in zijn geheel op het scherm overgenomen, druk dan op:

```
CTRL/STOP
```

Alle 'aangeklikte' coördinaten komen in beeld en de vraag OPSLAAN ? (J/N) verschijnt. Als er een fout gemaakt is, typen we N en beginnen gewoon opnieuw. Is alles goed gegaan, dan zetten we de bandrecorder op opname (SAVE/REC) en beantwoorden de vraag met J. De getekende lijn wordt opgeslagen. Het programma vraagt vervolgens of we een nieuwe lijn willen toevoegen en start zichzelf opnieuw. We bewegen de tekenpunt naar het begin van een volgende lijn. Op deze wijze werken we alle lijnen afzonderlijk af. Deze methode heeft als voordeel dat een bepaald lijnstuk altijd op het bandje terug is te vinden en, indien nodig, ook achteraf nog kan worden veranderd. Na het laatste lijnstuk zet de computer een stopcode op de band (het getal 300).
Begin niet gelijk het plafond van de Sixtijnse kapel te tekenen. Maak bijvoorbeeld een plattegrond van uw huis. Of gewoon een cirkel. Een lijn mag maximaal tweehonderd punten bevatten. U zult ze maar zelden allemaal nodig hebben. Mocht het gebeuren dat u ze toch heeft verbruikt, dan merkt u dat de pijl-

besturing niet meer werkt. Gebruik CTRL/STOP en J om het afgewerkte lijndeel op te slaan maar onthoud even waar u gebleven was. Het resterende deel wordt dan als een nieuwe lijn ingevoerd.
Nu de eerste tekening gemaakt is, willen we deze ook weer vanaf het bandje in de computer terug hebben. Daartoe dient het volgende, wel zeer eenvoudige, programma.

```
10  REM tekening van tape naar scherm
20  CLS
30  PRINT"Tape klaar... druk op een toets"
40  A$=INPUT$(1)
50  DIM X(200):DIM Y(200)
60  SCREEN 2:COLOR 1,10,10:CLS
70  OPEN"CAS:" FOR INPUT AS #1
80  INPUT#1,Q
90  IF Q=300 THEN 170
100 FOR I=1 TO Q
110   INPUT#1,X(I),Y(I)
120   IF I<2 THEN 140
130   LINE(X(I-1),Y(I-1))-(X(I),Y(I))
140 NEXT I
150 CLOSE
160 GOTO 70
170 CLOSE
180 RUN"CAS:"                (180 GOTO  180)
```

Zorg er bij het testen voor dat de cassetterecorder aanstaat (op afspelen) en de band aan het begin van de opgenomen lijnstukken. Lijn voor lijn wordt de tekening op het beeldscherm opnieuw in elkaar gezet. Een prachtige show omdat de computer zelf de recorder start en stopt om het volgende lijnstukje op te halen.
Als we een tekening als pareltje aan de ketting willen rijgen, dan zetten we eerst het zojuist beschreven uitvoerprogramma op de band. Direct daar achteraan komt de tekening (de tape-uitvoer van het tekenprogramma). Het bandje moet dus een keer worden verwisseld om het tekenprogramma, dat op een andere band staat, te kunnen laden. Als het uitvoerprogramma alle informatie van de band heeft gehaald, treft het vanzelf de volgende parel aan, die door regel 180 wordt geladen en gestart. Voor elke opgeslagen tekening komt dus het uitvoerprogramma te staan.
De coördinatenparen kunnen ook op papier worden afgedrukt. We volstaan met een eenvoudig programma. De getallenrij kan dan, zoals in het Lorelei-programma, met de hand als data aan een programma worden toegevoegd. Iets meer werk, dat wel. Een handige programmeur zal er niet veel moeite mee hebben het volgende programma zodanig te wijzigen dat het programma zelf de DATA-regels produceert. Als voorbeeld kan de sprite-ontwerper uit hoofdstuk 15 worden gebruikt. Dat programma genereert automatisch de benodigde DATA-regels.

```
10  REM co-ordinatenuitvoer naar printer
20  CLS:KEY OFF
30  PRINT"Tape klaar... druk op toets"
40  A$=INPUT$(1)
50  CLS
60  PRINT"Printer klaar... druk op toets"
70  A$=INPUT$(1)
80  CLS
90  DIM X(200):DIM Y(200)
100 LOCATE 9,10:PRINT "WACHT OP UITVOER"
110 OPEN"CAS:" FOR INPUT AS #1
120 INPUT#1,Q
130 IF Q=300 THEN CLOSE:END
140 LPRINT
150 FOR I=1 TO Q
160   INPUT#1,X(I),Y(I)
170   LPRINT X(I);Y(I);
180 NEXT I
190 CLOSE
200 LPRINT:LPRINT
210 GOTO 110
```

Op de printer verschijnen de X- en Y-coördinaten voor elk lijnstuk in paren achter elkaar. De rij wordt per lijnstuk voorafgegaan door een getal dat aangeeft hoeveel coördinatenparen er in het desbetreffende lijnstuk aanwezig zijn. Dit getal gebruikt de computer om de afzonderlijke lijnstukken te kunnen herkennen.
Als de stroom getallen ingelezen en afgedrukt is, kan het programma op de normale manier worden onderbroken met CTRL/STOP. Het kan gebeuren dat de rij laatste getallen pas wordt afgedrukt als het programma al is gestopt. Zet de printer daarom niet voortijdig uit.
Afgezien van Lorelei hebben we in dit hoofdstuk geen pareltjes aan de ketting toegevoegd. We hebben nu echter wel het gereedschap om zelf zeer individuele pareltjes te ontwerpen. Computertekeningen die achteraf weer kunnen worden opgeroepen, zouden wel eens erg bruikbaar kunnen blijken te zijn bij instructieprogramma's, bij het ontwerpen van achtergronden voor spelletjes of voor het maken van grote (afwijkende) letters. De tekeningen kunnen zonder meer achteraf verder worden bewerkt. Zo kunnen ze over het scherm worden verplaatst door bij de coördinaten een getal op te tellen of er een getal af te trekken. Ook is het achteraf inkleuren van de tekening mogelijk. Om dit laatste te demonstreren, geeft het volgende programma een letter weer waarvoor slechts weinig coördinaten nodig zijn. Dit om al te veel typewerk te voorkomen.

```
10  REM Assen
20  SCREEN 2:COLOR 1,10,10:CLS
30  FOR I=1 TO 9
40    READ X(I),Y(I)
50    IF I<2 THEN 80
```

```
60     LINE(X(I-1),Y(I-1))-(X(I),Y(I)),5
70     LINE(X(I-1)+70,Y(I-1)+70)-(X(I)+
       70,Y(I)+70),2
80   NEXT I
90   PAINT(30,1),5
100  PAINT(100,71),2
110  DATA 29,0,104,0,97,26,70,26,59,96,40,96,
          51,26,24,26,29,0
120  RUN"CAS:"                  (120 GOTO 120)
```

Eindelijk weer eens een programma dat aan de ketting kan. Het laat twee zaken zien. De DATA-regel bevat de coördinaten van een grote letter T. Regel 60 zorgt er dan ook voor dat op de opgegeven plaats een T op het scherm wordt gezet. Regel 70 echter verhoogt alle coördinaten met 70. Dezelfde letter T wordt dus 70 beeldpunten naar rechts en 70 beeldpunten naar beneden nogmaals afgebeeld. Bovendien gebeurt dit in een andere kleur. In de regels 90 en 100 krijgen de letters elk hun eigen kleurtje. Deze snelle letters kunnen nog sneller.

```
10   REM TT
20   SCREEN 2:COLOR 1,10,10:CLS
30   FOR I=1 TO 9
40     READ X(I),Y(I)
50     IF I<2 THEN 70
60     LINE(X(I-1),Y(I-1))-(X(I),Y(I))
70   NEXT I
80   PAINT(30,1)
90   FOR I=1 TO 9
100    IF I<2 THEN 140
110    FOR T=1 TO 10
120      LINE(X(I-1)+10*T,Y(I-1)+5*T)-
         (X(I)+10*T,Y(I)+5*T),2
130    NEXT T
140  NEXT I
150  DATA 29,0,104,0,97,26,70,26,59,96,40,96,
          51,26,24,26,29,0
160  RUN"CAS:"                  (160 GOTO 160)
```

Ze worden nu in volle vaart op het scherm herhaald. Het teken- en het weergeefprogramma uit dit hoofdstuk bieden zoveel grafische mogelijkheden dat het de moeite waard is ze op een afzonderlijk bandje te bewaren. Op datzelfde bandje kan ook de sprite-ontwerper uit het volgende hoofdstuk worden gezet. Een duidelijke sticker op de cassette zorgt ervoor dat dit bandje met grafische hulpgereedschappen altijd gemakkelijk te herkennen en snel terug te vinden is.

14 Flitsende geesten

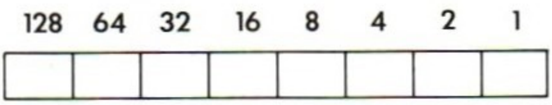
Sprites, het magisch toverwoord voor computerfanaten. Pacman, Zaxxon, Ghost Busters, noem maar op. Overal zien we ze, kleine figuurtjes die over het scherm bewegend grote vernielingen aanrichten, vitaminepillen eten of in afgronden storten. Sprites, moeilijk te begrijpen, moeilijk te maken en moeilijk te manipuleren. Alhoewel... niet in MSX-BASIC! Kijk zelf maar.

```
10  REM sprite
20  SCREEN 2,1:COLOR ,1,1:CLS
30  S$=""
40  FOR I=1 TO 8
50    READ A$
60    A=VAL("&H"+A$)
70    S$=S$+CHR$(A)
80  NEXT I
90  DATA 18,3C,24,7E,DB,7E,24,00
100 SPRITE$(0)=S$
110 X=120:Y=85
120 FOR T=1 TO 500
130   X=X+RND(1)*10-5
140   Y=Y+RND(1)*8-4
150   PUT SPRITE 0,(X,Y),6,0
160 NEXT T
170 CIRCLE(X+7,Y+7),20,5
180 RUN"CAS:"     (180 GOTO 180)
```

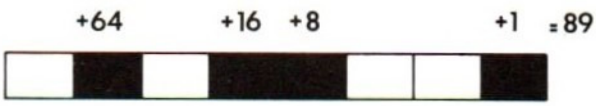
Waar komt die engerd nu weer vandaan? Van Mars of van Pluto, van een verre asteroïde? Nee hoor, gewoon uit regel 90. Hoe sprites worden opgebouwd is in veel verschillende boeken en MSX-handleidingen terug te vinden. Afhankelijk van de voorkeur voor getallenstelsels kunnen we sprites op drie manieren definiëren. In decimale getallen, in hexadecimale getallen (zoals in het voorgaande programma) en in binaire getallen.
Iedereen kan zo zijn of haar eigen voorkeur hebben. We zullen hier de binaire en de decimale schrijfwijze even kort herhalen aan de hand van de griezel uit het programma. Er zijn twee soorten sprites. Het verschil zit 'm uitsluitend in de afmeting. Er zijn sprites van 8 × 8 beeldpunten en er zijn sprites van 16 × 16 beeldpunten. De onderhavige geest is een 8 × 8 figuur. Binair is een sprite heel eenvoudig te beschrijven. Als we uitgaan van onze geest van 8 × 8 beeldpunten dan ziet het basisontwerp er als volgt uit:

```
---XX---    DATA ØØØ11ØØØ   ( 24)
--XXXX--    DATA ØØ1111ØØ   ( 60)
--X--X--    DATA ØØ1ØØ1ØØ   ( 36)
-XXXXXX-    DATA Ø111111Ø   (126)
XX-XX-XX    DATA 11Ø11Ø11   (219)
-XXXXXX-    DATA Ø111111Ø   (126)
--X--X--    DATA ØØ1ØØ1ØØ   ( 36)
--------    DATA ØØØØØØØØ   (  0)
```

De achter het figuurtje afgedrukte regels zijn de DATA-regels zoals die in het programma worden opgenomen als de binaire vorm wordt gebruikt. De getallen tussen haakjes geven de overeenkomstige decimale waarde aan. Die is eenvoudig af te leiden:

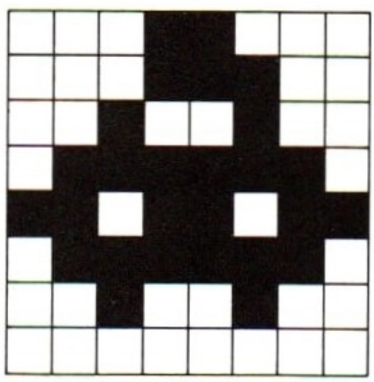


Voor elk vakje dat zwart wordt gemaakt, tellen we het erboven staande getal op. Voorbeeld:



Het zal duidelijk zijn dat de decimale ontwerpmethode het minste typewerk vraagt, maar dat de binaire methode een herkenbaar beeld oplevert. Tenminste, als de DATA-regels onder elkaar worden geplaatst. Vergelijk:

```
100 DATA 00011000
110 DATA 00011100
120 DATA 00100100
130 DATA 01111110
140 DATA 11011011
150 DATA 01111110
160 DATA 00100100
170 DATA 00000000
```

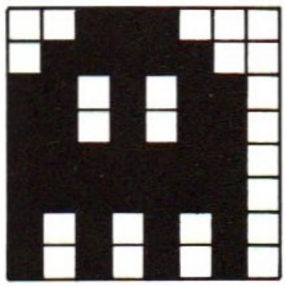


met:

```
100 DATA 24,28,36,126,219,126,36,0
```

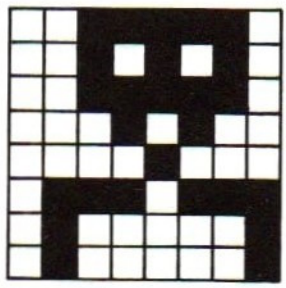
Vanwege de toch wel erg compacte schrijfwijze zullen we ons verder beperken tot de decimale sprite-notatie.
Een sprite laat zich het eenvoudigst ontwerpen op een vel ruitjespapier. Dat kan ook zelf getekend worden. Maak de hokjes die te zamen een SPRITE moeten vormen zwart en bekijk het resultaat vanaf enige afstand. Dan krijgt u een goed beeld van hoe de sprite er op het scherm uit zal gaan zien. Het is ook handig om vierkantjes uit zwart karton te knippen, zodat er gemakkelijk wat mee kan worden geschoven. Een nog betere methode om sprites te ontwerpen en visueel te beoordelen, wordt verderop in dit hoofdstuk beschreven. Eerst zullen we een paar sprites maken en op het scherm toveren. Het meest dankbaar zijn de spookachtige figuurtjes, maar ook vliegtuigen en race-auto's zijn geliefd.

Een geest om bang van te worden:



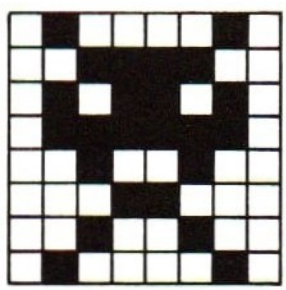
```
DATA 56,124,214,214,254,254,17Ø,17Ø
```

Zijn compagnon:



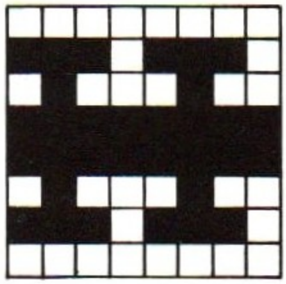
```
DATA 62,42,62,2Ø,8,119,65,65
```

En hun baas:



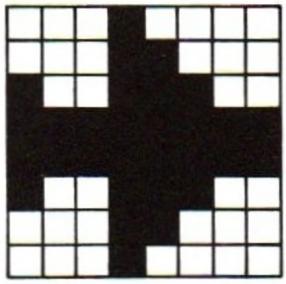
```
DATA 66,6Ø,9Ø,126,36,24,36,66
```

Een race-auto:



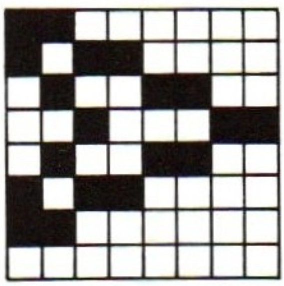
```
DATA 0,238,68,255,255,68,238,0
```

Een vliegtuig:



```
DATA 16,24,156,255,255,156,24,16
```

En tot slot een raket:



```
DATA 192,176,76,35,76,176,192,0
```

Dit zijn allemaal heel eenvoudige sprites. Toch kunnen we er al aardige dingen mee doen. Voor we zover zijn, bekijken we hoe een sprite in een programma wordt gebruikt. Elke sprite krijgt een eigen naam en wordt opgebouwd uit een string. In die string komen de DATA-codes te staan. Omdat die codes uit getallen bestaan, moeten we die eerst omzetten in een string.
Voor de raket-sprite ziet het geheel er dan zó uit:

```
SPRITE$(1)=CHR$(192)+CHR$(176)+CHR$(76)+CHR$(35)+
           CHR$(76)+CHR$(176)+CHR$(192)+CHR$(0)
```

Dat ziet er niet als erg aantrekkelijk typewerk uit. Het is beter de computer zelf voor zijn zaakjes te laten zorgen. Een programma dat zelf de DATA-regel leest, er een sprite van maakt en die op het scherm zet, heeft voor decimale DATA de volgende vorm:

```
10  REM geest
20  SCREEN 2,1:COLOR,1,1:CLS
30  FOR I=1 TO 8
40    READ S
50    S$=S$+CHR$(S)
60  NEXT I
70  SPRITE$(1)=S$
80  PUT SPRITE 0,(120,100),10,1
90  GOTO 90
100 DATA 56,124,214,214,254,254,170,170
```

Enkele opmerkingen kunnen verhelderend zijn. Het nummer van de sprite mag van 0 tot 256 lopen. Let wel, dat geldt voor de kleintjes waar we momenteel mee bezig zijn. Verderop komt het echte werk met 16×16 sprites, waarvan er 'slechts' 64 verschillende voor mogen komen.
In de PUT SPRITE-opdracht staat het sprite-nummer helemaal achteraan. Het eerste nummer heeft betrekking op het vlak waarin de sprite acteert.
Er zijn 31 vlakken achter elkaar mogelijk. Een sprite in vlak 1 gaat op het scherm ook echt achter een sprite in vlak 0 langs. Een sprite in het laatste vlak (31) schuift achter alle andere sprites langs.
Een paar spookjes zoals die in dit hoofdstuk beschreven zijn, treden op in het geestenprogramma.

```
10  REM spokendans
20  SCREEN 2,0:COLOR ,4,4:CLS
30  FOR I=1 TO 8:READ S:S$=S$+CHR$(S):NEXT I
40  FOR I=1 TO 8:READ G:G$=G$+CHR$(G):NEXT I
50  SPRITE$(0)=S$
60  SPRITE$(1)=G$
70  X=0:Y=0
80  FOR I=1 TO 15 STEP .2
90    X=X+RND(1)*10-RND(1)*5
100   Y=Y+RND(1)*5-RND(1)*2
110   X1=30*COS(I):X2=2*X1
120   Y1=30*SIN(I):Y2=2*Y1
130   PUT SPRITE 2,(X,Y),11,0
140   PUT SPRITE 1,(X+X1,Y+Y1),9,1
150   PUT SPRITE 0,(X+X2,Y+Y2),14,1
160 NEXT I
170 DATA 56,124,214,214,254,254,170,170
180 DATA 62,42,62,20,8,119,65,65
190 RUN"CAS:"                   (190 GOTO 190)
```

De inmiddels overbekende laatste regel rijgt de spoken aan de rode draad die door dit boek loopt.
De behandelde sprites zijn weliswaar leuk, maar toch verre van spectaculair. We kunnen er iets aan doen door ze wat op te blazen. Het enige wat we daarvoor aan het programma hoeven te wijzigen, is de SCREEN-opdracht in regel 20. Probeer maar eens:

```
20 SCREEN 2,1:COLOR,4,4:CLS
```

Het formaat scheelt aanzienlijk.
De SCREEN-opdracht kent als tweede variabele een getal dat bepaalt welke sprites worden geaccepteerd (8 × 8 of 16 × 16) en de afmeting waarop ze afgebeeld dienen te worden.

SCREEN 2,Ø	8 × 8 normaal
SCREEN 2,1	8 × 8 tweemaal vergroot
SCREEN 2,2	16 × 16 normaal
SCREEN 2,3	16 × 16 tweemaal vergroot

Jammer dat de sprites met verschillende afmetingen niet door elkaar te gebruiken zijn. Om de allermooiste sprites te kunnen maken, is het 16×16-formaat ideaal. Het ontwikkelen van sprites met deze afmeting is echter heel wat lastiger dan bij hun kleinere broertjes het geval is. Ook hier komt de computer te hulp. In het volgende hoofdstuk wordt een geraffineerd ontwerpprogramma besproken dat ons in staat stelt voor elke mogelijke vorm de best lijkende sprite te ontwerpen. Het lijkt wel wat op het tekenprogramma uit hoofdstuk 13.

15 Sprite-ontwerper

In tegenstelling tot de meeste programma's uit dit deel van het boek is het sprite-ontwerpprogramma tamelijk uitgebreid. Het presteert dan ook een en ander. Het tekent een patroon van zestien bij zestien velden. Hierin wordt met de pijlbesturing het te vullen vlak aangewezen en met een druk op de spatiebalk wordt het aangewezen vlak zwart gekleurd. Indien we een verkeerd vlak hebben gevuld, kunnen we dit vlak weer 'leeg' maken door het nogmaals met de spatiebalk aan te klikken. Als de sprite compleet is, berekent het programma de decimale getallen die de sprite vastleggen. Bovendien genereert het programma zelfstandig twee DATA-regels waarin de getallen staan. Deze kunnen ook op papier worden weergegeven. De DATA-regels die op het scherm worden afgebeeld, zijn al van regelnummers voorzien. Deze kunnen als afzonderlijke programma-onderdelen worden opgeslagen op een cassettebandje. Zo kunt u een aardige sprite-collectie opbouwen. We kunnen er ook andere programmaregels aan toevoegen die de zojuist ontworpen sprite een eigen leven laten leiden (lijden).
Voordat u dit programma test, moet u het wel 'saven'. Aan het einde van het programma wordt het programma gewist, opdat alleen de DATA-regels met de sprite-gegevens in het geheugen van de computer blijven staan.

```
10  REM ** 16x16 Sprite-ontwerper ***
20  KEY OFF:SCREEN 0:COLOR15,4,4:CLS
30  PRINT"Sprite-afmeting"
40  LOCATE 10,10:PRINT"1 Klein (Scherm 2)"
50  LOCATE 10,12:PRINT"2 Groot (Scherm 3)"
60  A$=INPUT$(1):A=VAL(A$)
70  IF A<1 OR A>2 THEN 20
80  A=A+1
90  SCREEN 2,A:COLOR 1,4,4:CLS
100 DIM V(2,31):DIM C(8,31)
105 REM **Bij einde sprite-ontwerp: 600
110 STOP ON
120 ON STOP GOSUB 600
125 REM **Ontwerpstramien-omlijsting
130 OPEN"GRP:" FOR OUTPUT AS #1
140 PSET(40,0),4
150 COLOR 15
160 PRINT#1,"SPRITE ONTWERP"
170 COLOR 1
175 REM **Ontwerpstramien-raster
180 FOR I=1 TO 17
190   LINE(32,32+I*8)-(160,32+I*8)
200   LINE(24+I*8,40)-(24+I*8,168)
210   LINE(96,32)-(96,176)
220 NEXT I
```

```
230 PSET(166,0),1:LINE -STEP(75,50),1,BF
235 REM **Cursor-besturing met pijltoetsen
240 XN=36:YN=44
250 PSET(XN,YN)
260 X=XN:Y=YN
270 A=STICK(0)
280 XN=X+8*(A=6)+8*(A=7)+8*(A=8)-8*(A=2)-
    8*(A=3)-8*(A=4)
290 YN=Y+8*(A=2)+8*(A=1)+8*(A=8)-8*(A=4)-
    8*(A=5)-8*(A=6)
295 REM **Rand van het ontwerpraster
300 IF XN<36 THEN XN=36
310 IF XN>156 THEN XN=156
320 IF YN<44 THEN YN=44
330 IF YN>164 THEN YN=164
335 REM **Cursor op de goede plaats zetten
340 PRESET(X,Y):PSET(XN,YN)
350 PRESET(X-1,Y-1):PSET(XN-1,YN-1)
360 PRESET(X+1,Y+1):PSET(XN+1,YN+1)
370 PRESET(X+1,Y-1):PSET(XN+1,YN-1)
380 PRESET(X-1,Y+1):PSET(XN-1,YN+1)
390 IF STRIG(0) THEN GOSUB 410
392 REM **Spatiebalk ingedrukt?
394 REM   Ja: subroutine vul hokje
396 REM  Nee: pijltoetsen uitlezen
400 GOTO 260
405 REM **Subroutine vul hokje
410 PAINT(XN+1,YN)
420 X=(XN-36)/8:Y=(YN-44)/8
430 IF X>=8 THEN A=2:X1=X-8:Y1=Y+16
            ELSE A=1:X1=X:Y1=Y
432 REM **Plaatsbepaling gevuld hokje
434 REM   en bepalen of hokje al gevuld
436 REM   is. Zo ja, hokje weer
438 REM   leegmaken(530)
440 IF C(X1,Y1)<>0 THEN GOSUB 530
                   ELSE 570
445 REM **String-uitdrukking voor sprite
450 S$=""
460 FOR I=1 TO 2
470   FOR J=0 TO 15
480     S$=S$+CHR$(V(I,J))
490 NEXT J,I
500 SPRITE$(0)=S$
504 REM **Sprite op het voorbeeldscherm
506 REM   zetten
510 PUT SPRITE 0,(185,5),10,0
515 REM **Terug naar cursorbesturing
520 RETURN
524 REM **Gevuld hokje weer leegmaken
526 REM   en "sprite-getalen" aanpassen
530 C(X1,Y1)=0
540 V(A,Y)=V(A,Y)-2^(7-X1)
550 LINE(XN-4,YN-4)-(XN+4,YN+4),4,BF
```

```
555 LINE(XN-4,YN-4)-(XN+4,YN+4),1,B
560 RETURN
565 REM **"Sprite-getallen" aanpassen
570 C(X1,Y1)=2^(7-X1)
580 V(A,Y)=V(A,Y)+2^(7-X1)
590 RETURN
600 REM **Uitvoer
605 REM **Presentatie sprite-getallen
610 PSET(166,0),1:LINE -STEP(75,180),1,BF
620 FOR I=0 TO 15
630   PSET(170,44+8*I),4:COLOR 10:
      PRINT #1,V(1,I)
640   PSET(210,44+8*I),4:COLOR 10:
      PRINT #1,V(2,I);
650   COLOR 1
660 NEXT I
670 PSET(0,0),4:COLOR 10:PRINT#1,
    STRING$(22,219)
680 PSET(0,0),4:COLOR 1:PRINT#1,
    "Printer uitvoer (J/N)"
690 A$=INPUT$(1)
700 IF A$="J" OR A$="j" THEN 710 ELSE 780
705 REM **Uitvoer sprite-getallen op papier
710 LPRINT"DATA";
720 FOR I=0 TO 14:LPRINT V(1,I);",";:NEXT I
730 LPRINT V(1,15)
740 LPRINT"DATA";
740 FOR I=0 TO 14:LPRINT V(2,I);",";:NEXT I
760 LPRINT V(2,15)
770 CLOSE
775 REM **Uitvoer sprite-getallen op scherm
780 SCREEN 0:COLOR 15,4,4:CLS
790 PRINT"8000 DATA";
800 FOR I=0 TO 14
802   PRINT STR$(V(1,I))+",";
804 NEXT I
810 PRINT STR$(V(1,15)):PRINT
820 PRINT"8010 DATA";
830 FOR I=0 TO 14
832   PRINT STR$(V(2,I))+",";
834 NEXT I
840 PRINT STR$(V(2,15)):PRINT
850 LOCATE 0,10
860 PRINT"Plaats cursor op regelnummers"
870 PRINT"en druk op <RETURN>"
880 PRINT
882 REM **Verwijdering programmaregels om
884 REM   in het geheugen alleen de sprite-
886 REM   getallen over te houden
890 PRINT"Plaats cursor op:"
900 PRINT"DELETE 10-870"
910 PRINT"_______________"
920 PRINT"en druk op <RETURN>"
930 PRINT
```

```
940 PRINT"Zet op cassette met:"
950 PRINT"SAVE"
960 REM **Einde programma
```

Door de cursor op de afgebeelde regel:

```
DELETE 1Ø-87Ø
```

te zetten, wordt het ontwerpprogramma verwijderd en resteren alleen nog de DATA-regels.
De sprite-ontwerper geeft de DATA-regels hoge nummers. Deze regels kunnen op cassette worden gezet met dezelfde opdracht waarmee alle in dit boek voorkomende programma's worden opgenomen:

```
SAVE"sprite"
```

Ze worden dan in de onverkorte ASCII-code weggeschreven en kunnen met MERGE"sprite" aan elk willekeurig programma worden toegevoegd.
De uitvoer van het programma kan er bijvoorbeeld zó uitzien:

```
8000 DATA 1,2,4,8,1618,21,23,21,18,8,4,
          4,2,2,2
8010 DATA 192,32,16,8,4,36,84,116,84,36,
          136,144,16,160,16,0,160
```

Hieronder staat het programma om deze geheimzinnige geest zichtbaar te maken.

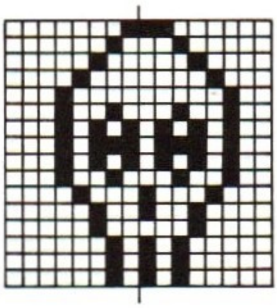
```
10 REM kleine 16x16 sprite
20 SCREEN 2,2:COLOR ,1,1:CLS
30 FOR I=1 TO 32:READ S:S$=S$+CHR$(S):NEXT I
40 SPRITE$(0)=S$
50 PUT SPRITE 0,(120,80),10,0
60 RUN"CAS:"                       (60 GOTO 60)
8000 DATA 1,2,4,8,16,18,21,23,21,
          18,8,4,4,2,2,2
8010 DATA 192,32,16,8,4,36,84,116,84,
          36,136,144,16,160,160,160
```

Voor de vergrote versie hoeft slechts één cijfertje in de SCREEN-opdracht te worden veranderd.

```
20 SCREEN 2,3:COLOR ,1,1:CLS
```

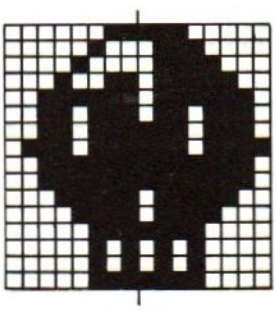
Ook kan die verandering in een programma plaatsvinden.

```
10   REM kleine en grote 16x16 sprite
20   FOR I=1 TO 32:READ S:
     S$=S$+CHR$(S):NEXT I
30   FOR J= 1 TO 5
40     FOR SC=2 TO 3
50       SCREEN 2,SC:COLOR ,1,1:CLS
60       SPRITE$(0)=S$
70       PUT SPRITE 0,(120,80),10,0
80       FOR Q=1 TO 300:NEXT Q
90   NEXT SC,J
100  RUN"CAS:"          (100 GOTO 100)
8000 DATA 1,2,4,8,16,18,21,23,21,18,
          8,4,4,2,2,2
8010 DATA 192,32,16,8,4,36,84,116,84,
          36,136,144,16,160,160,160
```

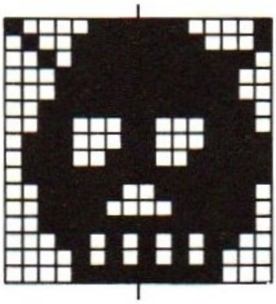


Drie andere sprites worden afgebeeld met de bijbehorende DATA-regels. Door de DATA-regels in het voorgaande programma te vervangen, doemen ze op uit de duisternis.

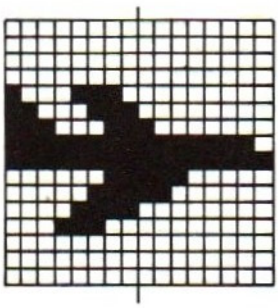
```
8000 DATA 1,2,4,9,31,31,29,29,61,31,31,
          15,7,2,2,3
8010 DATA 192,96,112,120,124,124,92,220,
          222,252,124,120,240,160,160,224
```



```
8000 DATA 135,79,63,31,63,56,120,121,127,
          127,62,63,63,26,10,15
8010 DATA 241,249,254,252,254,142,143,159,
          255,127,62,254,254,172,168,248
```



```
8000 DATA 0,0,0,0,135,195,225,255,255,3,7,
          15,28,0,0,0
8010 DATA 0,0,0,0,0,128,192,254,255,224,
          192,0,0,0,0,0
```



16 Spokendans en meer van dat moois

Sprites kunnen zich op 31 doorzichtige vlakken bevinden die achter elkaar staan. Ze kunnen dan ook achter elkaar langs bewegen.

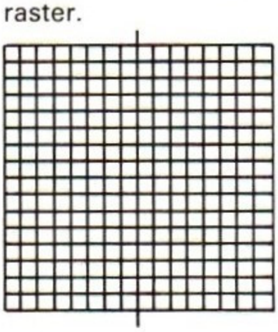
```
10   REM spokendans
20   SCREEN 2,3:COLOR ,1,1:CLS
30   FOR I=1 TO 32:READ S:S$=S$+CHR$(S):NEXT I
40   SPRITE$(0)=S$:SPRITE$(1)=S$:SPRITE$(2)=S$
50   FOR I=1 TO 515
60     PUT SPRITE 0,(I,70),10,0
70     PUT SPRITE 1,(256-I,80),7,1
80     PUT SPRITE 2,(2*I,90),9,2
90   NEXT I
100  RUN"CAS:"                    (100  GOTO 100)
8000 DATA 1,2,4,9,31,31,29,29,61,31,31,15,7,2,2,3
8010 DATA 192,96,112,120,124,124,92,220,222,252
8020 DATA 124,120,240,160,160,224
```

Heel fraai demonstreren de demonen hoe ze zich achter elkaar kunnen verschuilen. Er is nog iets bijzonders op het scherm te zien. Een geest verdwijnt helemaal van het scherm, voordat hij aan de andere kant weer tevoorschijn komt. Het MSX-scherm heeft aan de randen een uitloopje, vergelijkbaar met de coulissen van een toneelpodium.

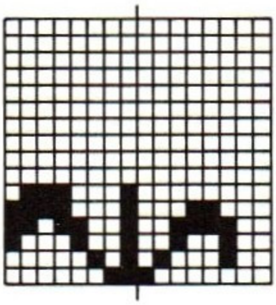
Bewegen de geesten wat beverig? Laat dan regel 90 eens weg. Supersnelle geesten flitsen dan over het scherm.

Doordat sprites zich op 31 schermen kunnen bevinden, kunnen we ook meerkleuren-sprites maken. We zetten er dan een aantal op dezelfde plaats. Er is slechts één beperking: nooit meer dan vier sprites op een regel. De sprites met de hoogste schermnummers worden aangetast of weggelaten als er meer dan vier zijn.

Sprites hoeven echt niet altijd uit een duistere wereld te komen. Ook in een tuintje doen ze het uitstekend. Als voorbeeld van een meerkleuren-sprite zullen we een bloemetje ontwerpen. Omdat een meerkleuren-portretje van een bloem uit meerdere delen is opgebouwd, gebruiken we een sprite-ontwerpraster.

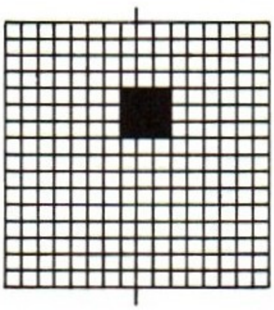


Als we een stapeltje fotokopieën van zo'n raster van 16 bij 16 hokjes maken, hoeven we slechts éénmaal een tekening te maken. Met kleurpotlood kan elk ingewikkeld ontwerp op redelijk eenvoudige wijze in elkaar worden gezet. Bij het maken van de sprite met behulp van het ontwerpprogramma is zo'n voorontwerp op papier een prachtig hulpmiddel. We beginnen met de groene bladeren van de bloem.



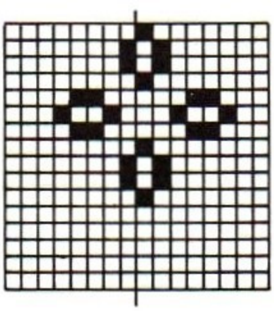
```
DATA Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,113,241,217,137,5,3
DATA Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,24,6Ø,36,68,128
```

Dan is het hartje van de bloem aan de beurt:



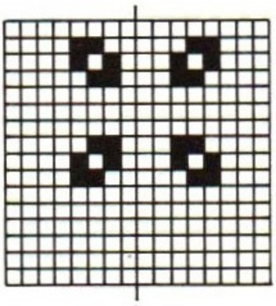
```
DATA Ø,Ø,Ø,Ø,1,1,1,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
DATA Ø,Ø,Ø,Ø,192,192,192,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø
```

De horizontale en verticale bloemblaadjes zien er zó uit:



```
DATA Ø,1,1,Ø,12,18,12,Ø,1,1,Ø,Ø,Ø,Ø,Ø,Ø
DATA 128,64,64,128,24,36,24,128,64,64,128,Ø,Ø,Ø,Ø,Ø
```

De diagonale bloemblaadjes completeren het geheel:



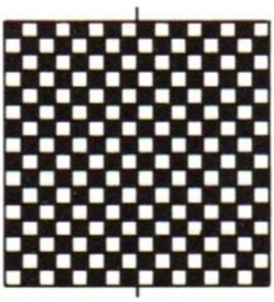
```
DATA Ø,12,1Ø,6,Ø,Ø,Ø,6,1Ø,12,Ø,Ø,Ø,Ø,Ø,Ø
DATA Ø,24,4Ø,48,Ø,Ø,Ø,48,4Ø,24,Ø,Ø,Ø,Ø,Ø,Ø
```

Het zojuist ontworpen bloemetje wordt in een pareltje gemonteerd:

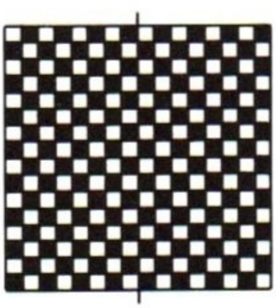
```
10   REM bloem
20   CLEAR 300
30   SCREEN 2,3:COLOR ,2,10:CLS
40   CIRCLE(85,70),40,3
50   PAINT(85,70),3
60   FOR J=0 TO 3
70     FOR I=1 TO 32
80       READ S:S$=S$+CHR$(S)
90     NEXT I
100    SPRITE$(J)=S$
110    S$=""
120  NEXT J
130  FOR I=1 TO 60
140    PUT SPRITE 3,(I*1.3,I),12,0
150    PUT SPRITE 2,(I*1.3,I),9,1
160    PUT SPRITE 1,(I*1.3,I),4,2
170    PUT SPRITE 0,(I*1.3,I),5,3
180    FOR Q=1 TO 100:NEXT Q
190  NEXT I
200  RUN"CAS:"         (200 GOTO 200)
8000 DATA 0,0,0,0,0,0,0,0,0,0,113,241,217,137,5,3
8010 DATA 0,0,0,0,0,0,0,0,0,0,24,60,36,68,128
8020 DATA 0,0,0,0,1,1,1,0,0,0,0,0,0,0,0
8030 DATA 0,0,0,0,192,192,192,0,0,0,0,0,0,0,0,0
8040 DATA 0,1,1,0,12,18,12,0,1,1,0,0,0,0,0,0
8050 DATA 128,64,64,128,24,36,24,128,64,64,128,0,0
8060 DATA 0,0,0,0,12,10,6,0,0,0,6,10,12,0,0,0,0,0
8070 DATA 0,0,24,40,48,0,0,0,48,40,24,0,0,0,0,0,0
```

Dat kleurencombinaties het niet altijd goed doen, laat de bloem zien die onderweg is naar het perkje. Eenmaal in het perkje aangekomen, ziet het er allemaal

een stuk fleuriger uit. Lijkt het niet erg veel op een kunstig borduurwerkje? (Misschien is het niet eens een gek idee om met de MSX-computer borduurwerkjes te ontwerpen.)
Bij het weglaten van regel 20 verschijnt er een foutmelding op het scherm. Normaal gesproken reserveert het MSX-systeem een aantal geheugenplaatsen voor strings. Als die hoeveelheid niet voldoende is, kunnen we met CLEAR het aantal plaatsen opgeven dat wél toereikend is; in dit geval 300. Hoewel dit getal aan de hand van het programma uit te rekenen is, blijkt het vaak praktischer het gewoon te proberen. Verhoog het getal, beginnend bij tweehonderd (de normale toestand) steeds met 50 tot de foutmelding niet meer verschijnt.
Als we het bloemetje meermalen willen afbeelden, is het zaak ervoor te waken dat er geen twee bloemen op één regel komen. Een bloem bestaat immers uit vier sprites, het maximum voor een regel. Schuin onder elkaar kan dus wel.
Nu we toch met meerkleuren-sprites bezig zijn, is het aardig om het eerder besproken verfmengproces eens sprite-matig te bekijken. Stel we maken twee rasters, waarbij de beeldpunten om en om zijn ingevuld:



```
DATA 17Ø,85,17Ø,85,17Ø,85,17Ø,85,17Ø,85,17Ø,85,17,
     85,17Ø,85
DATA 17Ø,85,17Ø,85,17Ø,85,17Ø,85,17Ø,85,17Ø,85,17,
     85,17Ø,85
```

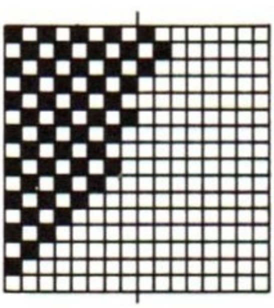


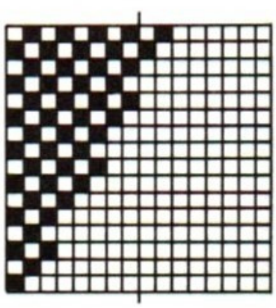
```
DATA 85,17Ø,85,17Ø,85,17Ø,85,17Ø,85,17Ø,85,17Ø,85,
     17Ø,85,17Ø
DATA 85,17Ø,85,17Ø,85,17Ø,85,17Ø,85,17Ø,85,17Ø,85,
     17Ø,85,17Ø
```

In een programma krijgen de rasters een eigen kleur en worden ze over elkaar heen geprojecteerd.

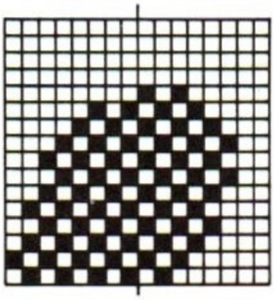
```
10   REM oranje sprite
20   SCREEN 2,2:COLOR,1,1:CLS
30   FOR J=0 TO 1
40     FOR I=1 TO 32
50       READ S:S$=S$+CHR$(S)
60     NEXT I
70     SPRITE$(J)=S$
80     S$=""
90   NEXT J
100 PUT SPRITE 0,(80,80),10,0
110 PUT SPRITE 1,(80,80),6,1
120 RUN"CAS:"                (120 GOTO 120)
8000 DATA 170,85,170,85,170,85,170,85,170,
          85,170,85,170,85,170,85
8010 DATA 170,85,170,85,170,85,170,85,170,
          85,170,85,170,85,170,85
8020 DATA 85,170,85,170,85,170,85,170,85,
          170,85,170,85,170,85,170
8030 DATA 85,170,85,170,85,170,85,170,85,
          170,85,170,85,170,85,170
```

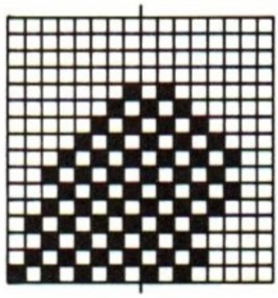
Een idee om eens te gebruiken, bijvoorbeeld in een spel met konijnen waarbij winterpeentjes het lokaas zijn. De kleur is er goed voor. Aan de afmeting ontbreekt echter iets.
Een hoogstandje van sprite-kunst zou een grote winterpeen zijn, in de kleur oranje, met nog een restje blad eraan. We gaan eerst weer met de rasters aan de slag. Omdat we lichtrood en donkergeel willen mengen, zetten we de beeldpunten om en om aan. Dat geldt voor de onderkant van de wortel:



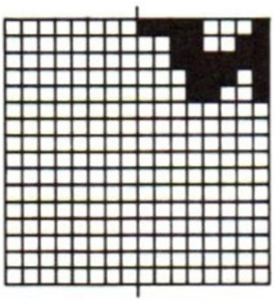


maar ook voor de bovenkant:





Voor het blad maken we de volgende sprite:



Een programma waarin de reuzenwortel een rol zonder happy end speelt, zou er uit kunnen zien als het volgende programma. Een vijfvoudige sprite in twee (eigenlijk drie) kleuren, dat is toch een mooi resultaat.

```
10    REM peen
20    SCREEN 2,3:COLOR 1,7,7:CLS
30    FOR J=0 TO 4
40      FOR I=1 TO 32
50        READ S:S$=S$+CHR$(S)
60      NEXT I
70      SPRITE$(J)=S$
80      S$=""
90    NEXT J
100   CIRCLE(60,110),30,11
110   PAINT(60,110),11
120   CIRCLE(45,70),30,11,,,6
130   PAINT(45,70),11
140   CIRCLE(62,70),30,11,,,6
150   PAINT(62,70),11
160   CIRCLE(61,100),8,1,,,2
170   PAINT(61,100),1
180   PSET(67,102),15
190   CIRCLE(-10,192),80,11
200   PAINT(1,191),11
210   CIRCLE(80,110),11,1,4,5
```

```
220  FOR I=1 TO 100
222    X=RND(1)*180:Y=192-RND(1)*50:R=RND(1)*20
230    CIRCLE(X,Y),R,2
235    PAINT (X,Y),2
240  NEXT I
250  FOR I=1 TO 120
260    PUT SPRITE 4,(200,I+32),9,0
270    PUT SPRITE 3,(200,I+32),10,1
280    PUT SPRITE 2,(200,I),9,2
290    PUT SPRITE 1,(200,I),10,3
300    PUT SPRITE 0,(200,I),12,4
310  NEXT I
320  FOR Q=1 TO 100:NEXT Q
330    PUT SPRITE 4,(200,208),9,0
340    PUT SPRITE 3,(200,208),10,1
350    PUT SPRITE 2,(200,208),9,2
360    PUT SPRITE 1,(200,208),10,3
370  PUT SPRITE 0,(58,115),12,4
380  RUN"CAS:"              (GOTO 380)

8000 DATA 170,85,170,85,170,84,170,84,168,80,
          168,80,160,64,128,128
8010 DATA 128,0,128,0,0,0,0,0,0,0,0,0,0,0,0,0
8020 DATA 85,171,85,170,85,170,84,168,84,168,
          80,160,64,160,64,128
8030 DATA 64,128,0,0,0,0,0,0,0,0,0,0,0,0,0,0
8040 DATA 0,0,0,0,0,5,10,21,42,21,42,85,42,85,
          170,85
8050 DATA 0,0,0,0,160,80,168,84,168,84,168,80,
          160,80,160,64
8060 DATA 0,0,0,0,1,2,5,10,21,42,21,42,85,
          42,85,170
8070 DATA 0,0,0,0,64,160,80,168,84,168,80,168,
          80,160,80,160
8080 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
8090 DATA 241,51,63,29,25,0,0,0,0,0,0,0,0,0,0,0
```

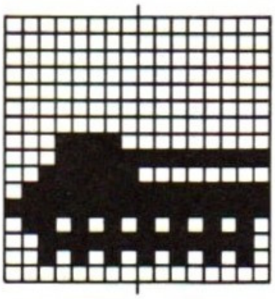
Zo'n uitgebreide DATA-lijst is niet echt leuk om in te typen. Het kleinste foutje is vaak al fataal. De DATA-lijsten kunnen ook door de sprite-ontwerper worden gemaakt. Bij meervoudige sprites veranderen we de door het ontwerpprogramma opgegeven regelnummers (8000 en 8010) in opeenvolgende nummers (8000-8010, 8020-8030 enz.). De regelparen 'saven' we als afzonderlijke programma's op een bandje. Achteraf kan er dan een programma van worden gemaakt met de MERGE-opdracht. Hiervoor is het nodig dat we niet de CSAVE- maar de SAVE-opdracht gebruiken. Daar zult u weinig moeite mee hebben, omdat we al onze pareltjes al met de SAVE-opdracht aan de ketting hebben geregen.

Er is nóg een aspect niet aan de orde geweest. In het beeldverhaal uit het vorige programma verdwijnt er plotseling een stuk wortel. Het zal iedereen duidelijk zijn wie daarvoor verantwoordelijk is. De eet-opdracht is te vinden in de regels 330 en 340. Als de Y-coördinaat van een sprite de waarde 208 of 209

heeft, verdwijnt hij van het scherm. Let hierbij op! Als de Y-coördinaat 208 is, verdwijnen ook alle sprites met een hoger schermnummer. Dat is niet alleen handig als verdwijntruc, een sprite kan er ook onheilspellend mee aan het knipperen worden gebracht.

```
10   REM ufo
20   SCREEN 2,3:COLOR 15,1,1:CLS
30   FOR J=0 TO 1
40     FOR I=1 TO 32
50       READ S:S$=S$+CHR$(S)
60     NEXT I
70     SPRITE$(J)=S$
80     SPRITE$(J+1)=S$
90     S$=""
100  NEXT J
110  FOR I=1 TO 80
120    PUT SPRITE 1,(I*1.3,I),12,0
130    PUT SPRITE 0,(I*1.3,I),4,1
140  NEXT I
150  FOR P=1 TO 50
160    PUT SPRITE 0,(I*1.3-1,I-2),9,2
170    FOR Q=1 TO 50:NEXT Q
180    PUT SPRITE 0,(I*1.3-1,209),9,2
190    FOR Q=1 TO 50:NEXT Q
200  NEXT P
210  RUN"CAS:"              (210 GOTO 210)
8000 DATA 192,32,16,9,7,63,127,230,102,
          63,31,19,33,0,0,0
8010 DATA 3,4,8,144,224,252,254,103,
          102,252,248,200,132,0,0,0
8020 DATA 0,0,0,0,0,0,0,0,25,25,0,0,
          0,0,0,0
8030 DATA 0,0,0,0,0,0,0,0,152,152,0,
          0,0,0,0,0
```

De sprites komen tot leven. Het zijn in het algemeen nogal agressieve wezentjes. Soms zó agressief dat ze geen waardig sieraad zijn aan de tot nu toe opgebouwde parelketting. Er is echter een onderdeel aan een parelketting waarbij de zacht glanzende schoonheid van de parel geen rol speelt: het slot. Dat wordt meestal onzichtbaar weggemoffeld. Als sluitstuk van dit deel van het boek zullen we wat bruut geweld aan het vorige programma toevoegen waarin dan toch het onvermijdelijke destructieve element naar voren komt. Als excuus voor het geweld op het scherm kan worden aangevoerd dat het een goede manier is om ON SPRITE GOSUB te demonstreren.
De computer kan zelf vaststellen of twee sprites met elkaar in botsing komen. We gaan dit proberen door de UFO die in het vorige programma voorkomt uit de lucht te schieten. In de eerste plaats moeten we hiervoor een tank het strijdtoneel op laten rijden.



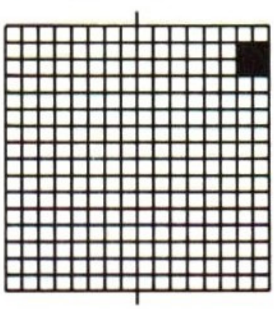
```
8040 DATA 0,0,0,0,0,0,0,30,63,63,127,255,
          106,63,42,0
8050 DATA 0,0,0,0,0,0,0,0,255,0,255,255,
          171,254,170,0
```

Als de gevechtspositie is ingenomen, richten we de loop op de UFO:



```
8060 DATA 0,0,0,0,0,0,0,30,63,63,127,255,
          106,63,42,0
8070 DATA 0,0,0,4,8,16,32,64,128,0,255,
          255,171,254,170,0
```

Nu nog het projectiel dat de loop op de juiste plaats verlaat:



```
8080 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
8090 DATA 0,3,3,0,0,0,0,0,0,0,0,0,0,0,0,0
```

De strijd tegen de indringer kan beginnen. Let op regel 20. We hebben de ruimte nodig dus gaan we naar SCREEN 2,2.

```
10    REM attack
20    SCREEN 2,2:COLOR 15,1,1:CLS
22    ON SPRITE GOSUB 1000
30    ON SPRITE GOSUB 1000
40    FOR J=0 TO 4
50      FOR I=1 TO 32
60        READ S:S$=S$+CHR$(S)
70      NEXT I
80      SPRITE$(J)=S$
90      S$=""
100   NEXT J
110   FOR I=1 TO 70
120     PUT SPRITE 1,(I*3,I),12,0
130   NEXT I
140   FOR P=1 TO 10
150     PUT SPRITE 0,(I*3-3,I-2),9,1
160     FOR Q=1 TO 50:NEXT Q
170     PUT SPRITE 0,(I*3-3,209),9,1
180     FOR Q=1 TO 50:NEXT Q
190   NEXT P
200   FOR I=1 TO 100
210     PUT SPRITE 2,(I,170),3,2
220     FOR Q=1 TO 30:NEXT Q
230   NEXT I
240   FOR Q=1 TO 300:NEXT Q
250   PUT SPRITE 2,(I-1,170),3,3
260   SPRITE ON
270   FOR Q=1 TO 300:NEXT Q
280   FOR I=1 TO 100
290     PUT SPRITE 4,(I+100,170-I),11,4
300   NEXT I
302   SPRITE OFF
310   RUN"CAS:"          (310 GOTO 310)

1000 FOR I=1 TO 50
1010   CIRCLE(220,80),RND(1)*30,RND(1)*14
1020 NEXT I
1030 SPRITE OFF
1040 RETURN

8000 DATA 192,32,16,9,7,63,127,230,102,63,
          31,19,33,0,0,0
8010 DATA 3,4,8,144,224,252,254,103,102,
          252,248,200,132,0,0,0
8020 DATA 0,0,0,0,0,0,0,0,25,25,0,0,0,0,0,0
8030 DATA 0,0,0,0,0,0,0,0,152,152,0,0,0,0,0,0
8040 DATA 0,0,0,0,0,0,0,30,63,63,127,255,106,
          63,42,0
8050 DATA 0,0,0,0,0,0,0,0,255,0,255,255,171,
          254,170,0
```

```
8060 DATA 0,0,0,0,0,0,0,30,63,63,127,255,106,
          63,42,0
8070 DATA 0,0,0,4,8,16,32,64,128,0,255,255,
          171,254,170,0
8080 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
8090 DATA 0,3,3,0,0,0,0,0,0,0,0,0,0,0,0,0
```

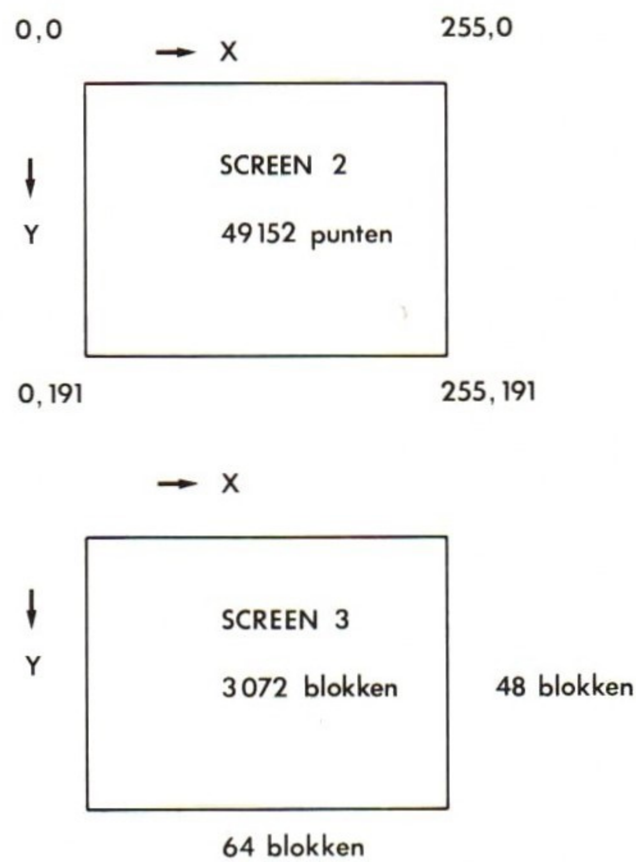
Wat de pareltjes aangaat: dit was het dan. Een parelsnoer vol interessante schermpjes. Sommige met beweging, een paar met geluid. Maar altijd een demonstratie van de onuitputtelijke grafische mogelijkheden van de MSX-computer. Een volwassen systeem binnen ieders bereik. Tussen het lezen van dit boek door heeft u veel typewerk verricht. Maar alles wat u heeft gedaan, ligt vast. Ga er eens rustig voor zitten en laat de parelketting nog eens langs het scherm glijden. Het is een indrukwekkend geheel geworden. Wellicht heeft u er al zelfgemaakte juweeltjes in opgenomen. Als het bandje de laatste parel heeft prijsgegeven, blijft het doelloos doorlopen. Wanhopig op zoek naar een volgend stukje werk. Uw werk !!!

17 Overzicht van de grafische commando's

SCREEN

```
SCREEN 2,Ø,1,Ø,Ø
      | | | | |
      | | | | Type printer
      | | | SAVE-snelheid
      | | Toetsaanslag klik aan/uit
      | Sprite-grootte
      Schermsoort
```

SCREEN 0 Tekstscherm 24 regels met 40 tekens.
SCREEN 1 Tekstscherm 24 regels met 32 tekens (groter lettertype).



Afbeelding 17-1 De opbouw van scherm 2 en scherm 3.

SCREEN 2 Scherm opgebouwd uit 192 horizontale rijen van 256 beeldpunten die onafhankelijk van elkaar kunnen worden aan- en uitgezet.
Beperking: Per groepje van acht horizontale beeldpunten mogen niet meer dan twee kleuren voorkomen.

SCREEN 3 Scherm opgebouwd uit 192 horizontale rijen van 256 beeldpunten die in groepjes van 4 × 4 beeldpunten onafhankelijk van elkaar kunnen worden aan- en uitgezet.

SCREEN ,0 Sprite-afmeting 8 × 8 beeldpunten.

SCREEN ,1 Sprite-afmeting 8 × 8 beeldpunten echter vergroot weergegeven.

SCREEN ,2 Sprite-afmeting 16 × 16 beeldpunten.

SCREEN ,3 Sprite-afmeting 16 × 16 beeldpunten echter vergroot weergegeven.

De drie laatste SCREEN-variabelen worden zelden gebruikt.

COLOR

```
COLOR 15,4,5
       | | |
       | | └─kleur van de schermrand
       | └─achtergrondkleur
       └─inktkleur
```

```
COLOR 15,4,5
```

Geeft witte letters op een donkerblauwe achtergrond waarbij de schermrand lichtblauw is.
Een achtergrondkleur verandert pas na een CLS-opdracht.

Kleurnummer	*Kleur*
0	transparant
1	zwart
2	groen
3	lichtgroen
4	donkerblauw
5	lichtblauw
6	donkerrood
7	cyaan
8	rood
9	lichtrood
10	donkergeel

Kleurnummer	*Kleur*
11	lichtgeel
12	donkergroen
13	paars
14	grijs
15	wit

PSET, PRESET

Werken uitsluitend bij SCREEN 2 en 3.

```
PSET (1ØØ,5Ø),4
      |   |   |
      |   |   └─kleur
      |   └─Y-coördinaat
      └─X-coördinaat
```

```
PSET (1ØØ,5Ø)
```

Laat het beeldpunt 100,50 oplichten in de geldende inktkleur.

```
PSET (1ØØ,5Ø),1Ø
```

Laat het aangegeven beeldpunt oplichten in de kleur geel.

```
PSET STEP(2Ø,1Ø),1Ø
```

Laat een punt geel oplichten dat 20 punten rechts van en 10 punten onder het laatst getekende beeldpunt ligt.
PRESET dooft een punt. Als PRESET van een kleur is voorzien, is er geen verschil met de PSET-opdracht.

LINE

Werkt uitsluitend bij SCREEN 2 en 3.

```
LINE (1Ø,2Ø)-(1ØØ,15Ø)
```

Trekt een lijn vanaf de beeldcoördinaten 10,20 naar het punt 100,150 in de van kracht zijnde inktkleur.

```
LINE (1Ø,2Ø)-(1ØØ,15Ø),6
```

Trekt dezelfde lijn in rood.

```
LINE (1Ø,2Ø)-(1ØØ,15Ø),,B
```

Tekent een rechthoek met de punten 10,20 en 100,150 als hoekpunten in de geldende inktkleur.

```
LINE (1Ø,2Ø)-(1ØØ,15Ø),12,BF
```

Tekent de rechthoek in de kleur donkergroen, maar vult hem bovendien op met de kleur donkergroen.

```
LINE -STEP(2Ø,2Ø)
```

Trekt een lijn vanaf het laatst getekende punt naar een punt dat 20 plaatsen naar rechts en twintig plaatsen naar beneden ligt.

```
LINE -STEP(2Ø,2Ø),6,BF
```

Tekent een donkerrood opgevuld vierkant met zijden van 20 × 20 beeldpunten. Het beginpunt is de laatst getekende beeldpunt.

CIRCLE

Werkt uitsluitend bij SCREEN 2 en 3.

```
CIRCLE (1ØØ,6Ø),3Ø,1Ø,1.5,3.5,1.35
         |    |   |   |  |   |   |
         |    |   |   |  |   |   └rondheid
         |    |   |   |  |   └eindhoek
         |    |   |   |  └beginhoek
         |    |   |   └kleur
         |    |   └straal
         |    └Y-coördinaat
         └X-coördinaat
```

```
CIRCLE (1ØØ,6Ø),3Ø
```

Tekent een cirkel met als middelpunt 100,60 en met een straal van 30 beeldpunten in de geldende inktkleur.

```
CIRCLE (1ØØ,6Ø),3Ø,1Ø
```

Tekent een gele cirkel.

```
CIRCLE (1ØØ,6Ø),3Ø,9,1.5,4.5
```

Tekent alleen de linkerhelft van de cirkel in de kleur lichtrood.

```
CIRCLE (1ØØ,6Ø),3Ø,9,,,1.35
```

Tekent een ronde cirkel. Het getal kan per monitor verschillend zijn. In het ideale geval is het getal 1.
De rondheid van de cirkel duidt de verhouding tussen X en Y aan. X en Y zijn resp. de horizontale en de verticale as van de cirkel.

```
CIRCLE STEP(2Ø,-1Ø),4Ø
```

Tekent een cirkel waarbij het middelpunt 20 beeldpunten rechts van en 10 beeldpunten boven het laatst getekende beeldpunt ligt.

PAINT

Werkt uitsluitend bij SCREEN 2 en 3.

```
PAINT (1ØØ,8Ø),6
        |    |   |
        |    |   └─kleur
        |    └─Y-coördinaat
        └─X-coördinaat
```

```
PAINT (1ØØ,8Ø)
```

Vult een in de geldende inktkleur getekende gesloten contour, waarbinnen het opgegeven punt ligt, op met de geldende inktkleur.

```
PAINT (1ØØ,8Ø),6
```

Vult een donkerrood getekende gesloten contour waarbinnen het opgegeven punt ligt donkerrood op.

```
PAINT STEP(-2Ø,1Ø),12
```

Gaat uit van een beeldpunt dat ten opzichte van het laatst getekende beeldpunt 20 plaatsen naar links en 10 plaatsen naar beneden is verschoven.

DRAW

Werkt uitsluitend bij SCREEN 2 en 3.
DRAW voert tekenopdrachten uit die worden gegeven in de vorm van een string.

```
DRAW "U6ØR4Ø"
```

Tekent een lijn 60 beeldpunten omhoog en vervolgens een lijn 40 beeldpunten naar rechts.

```
DRAW "BU6ØBR4Ø"
```

Het voorvoegsel 'B' zorgt ervoor dat er een verplaatsing optreedt zonder dat er een lijn wordt getrokken.

```
DRAW "NU6ØNR4Ø"
```

Het voorvoegsel 'N' laat de tekenpunt terugkeren naar het uitgangspunt nadat de lijn is getrokken.

```
DRAW "C6U6ØC1ØR4Ø"
```

Het voorvoegsel 'C' bepaalt de kleur van de te tekenen lijn.

```
DRAW "M1ØØ,8Ø"
```

Tekent een lijn vanaf het laatst getekende beeldpunt naar het punt 100,80.

```
DRAW "S8U6Ø"
```

Het voorvoegsel 'S' vergroot de schaal. De uitgangswaarde is S4. Het voorvoegsel S8 verlengt de lijn met een factor 2.

Deel 2
Het beeldbewerkingsproject

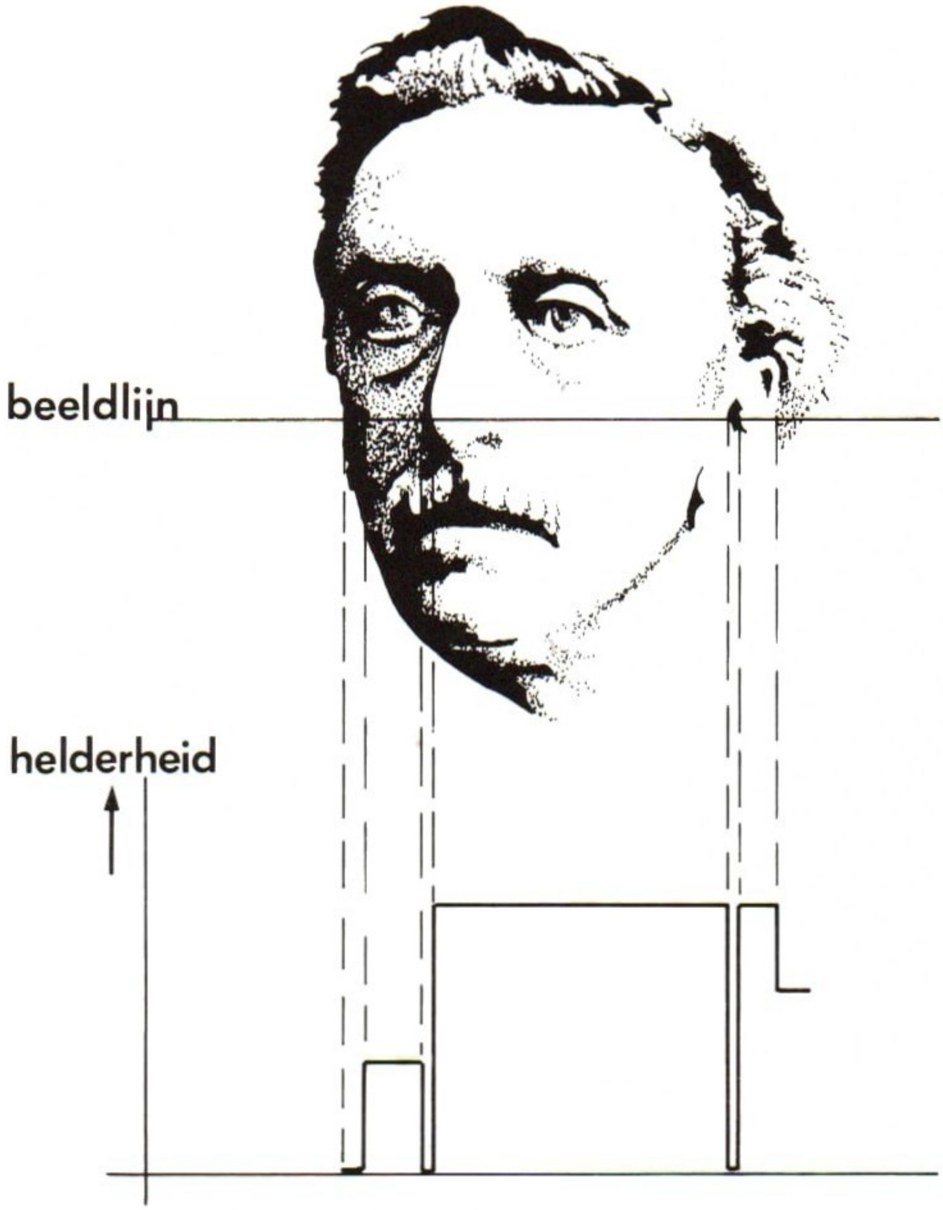


18 Inleiding deel 2

In dit deel gaan we samen op avontuur in de wondere wereld van 'advanced computer graphics'. We zullen kennismaken met veel aspecten van elektronische beeldvorming en de theorie ook echt in praktijk brengen. Hoewel de geschetste technieken op alle soorten computerbeelden toepasbaar zijn, geeft het bewerken van zelf-opgenomen beelden toch wel de grootste bevrediging. We beginnen dan ook met een eenvoudig elektronicaproject dat door iedereen uit te voeren is. In dit project zullen we de bouw beschrijven van een optische leeskop waarmee we zelf foto's, illustraties en andere afbeeldingen om kunnen zetten in een computerbeeld. Om het geheel ook voor de beginner geschikt te maken, zullen we moeilijke interfaceproblemen omzeilen door gebruik te maken van de joystick-ingang. Op deze wijze gaan we het risico dat schade wordt aangebracht aan het inwendige van de computer uit de weg. Hoewel het project geschikt is voor alle computers met een joystick-ingang, zijn de begeleidende programma's geschreven in MSX-BASIC. Het gebruik van speciale MSX-faciliteiten is echter zo veel mogelijk vermeden, zodat omzetting naar andere BASIC-dialecten in de meeste gevallen geen grote problemen zal opleveren.
Lees dit deel van het boek in de onmiddellijke nabijheid van de computer. De vele kleine programma's kunnen dan tijdens het lezen al worden uitgeprobeerd, zodat de werking ervan duidelijk is bij de behandeling van het complete beeldverwerkingsprogramma in het laatste hoofdstuk. Met dit programma kunnen beelden opgenomen, weergegeven, bewerkt en afgedrukt worden.

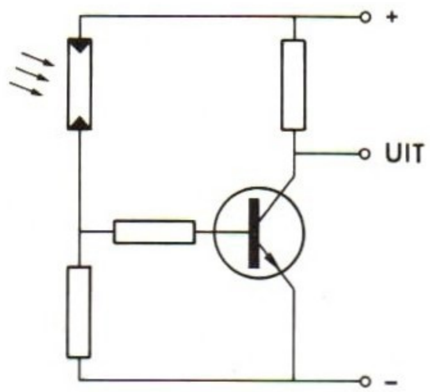
19 Het elektronicaproject

Dat de MSX-computer ongekende grafische mogelijkheden heeft, wordt door niemand ontkend.



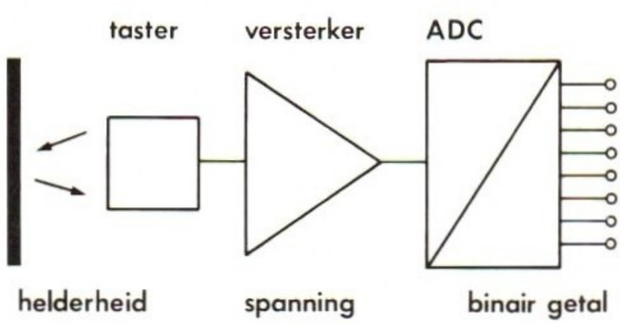
Afbeelding 19-1 De helderheid als functie van de plaats op een beeldlijn.

Toch kan het vullen van het grafisch scherm met een gedetailleerde afbeelding nog aardig wat problemen opleveren. Hoe zit dat dan met die fraaie computerbeelden waarop de wereld gedigitaliseerd weergegeven is? Of concreter gesteld: Hoe krijgen we een afbeelding vanaf papier rechtstreeks op het scherm ?
We duiken even in de TV-techniek. Dat het TV-beeld is opgebouwd uit lijnen mag als bekend worden verondersteld. Afbeelding 19-1 geeft zo'n lijn weer.
De helderheid langs de lijn varieert: alle grijstinten tussen zwart en wit kunnen worden afgebeeld. Het meten van de helderheid langs zo'n lijn is met wat elektronica nog wel te realiseren. Afbeelding 19-2 geeft hier een algemene opzet voor.



Afbeelding 19-2 Een algemene opzet voor het meten van de helderheid met behulp van een lichtgevoelige weerstand.

Als we de taster langs een lijn van een afbeelding bewegen, verschijnt er op de uitgang een signaal waarvan de hoogte afhankelijk is van de grijswaarde van de afbeelding. Een analoog signaal dus. Met een analoog/digitaalconverter kan dit signaal gedigitaliseerd worden, zodat het door de computer te verwerken is (zie afbeelding 19-3).



Afbeelding 19-3 Het omzetten van de helderheid in een getal dat door de computer te lezen is.

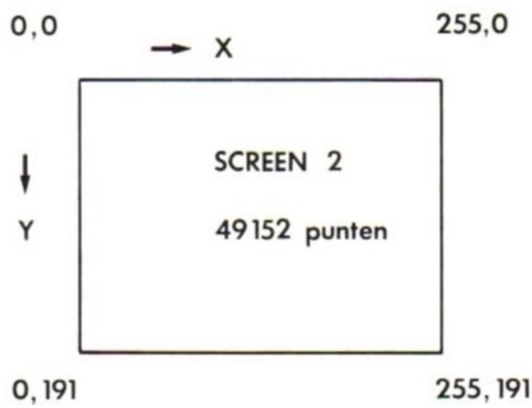
Wat ons te doen staat is:

- Het bouwen van een lichtopnemer die de intensiteit van het door de beeldpunten op papier gereflecteerde licht meet en omzet in een elektrisch signaal.
- Beslissen op welke wijze het signaal aan de computer wordt aangeboden.
- De bouw van een analoog/digitaalconverter (ADC) gebaseerd op de keuze uit het vorige punt.
- Het verzinnen van een aftastmechanisme om zodoende een afbeelding lijn voor lijn in het computergeheugen te kunnen opslaan.

Voor we met ongeremd enthousiasme met het ontwerpen van de diverse componenten beginnen, lijkt het niet onverstandig ons eisenpakket wat beter te omschrijven.
De activiteiten van de MSX-computer spelen zich af op vier mogelijke schermsoorten, waarbinnen zich weer een aantal varianten aftekenen ten aanzien van sprite-afmetingen.

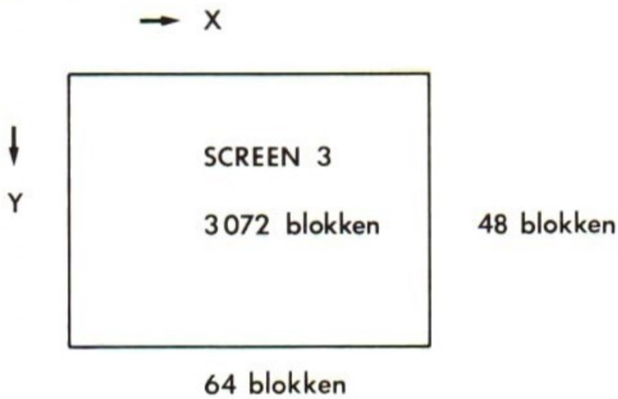
SCREEN 0	Een tekstscherm van 24 regels met elk 40 tekens.
SCREEN 1	Een tekstscherm van 24 regels met 32 iets bredere tekens.
SCREEN 2	Een scherm voor grafisch gebruik met een hoog oplossend vermogen van 256 bij 192 beeldpunten.
SCREEN 3	Een grafisch beeldscherm met een laag oplossend vermogen van 64 bij 48 beeldpunten.

Het zal duidelijk zijn dat we voor beeldvorming op de schermsoort 2 of 3 aangewezen zijn. Een nadere beschouwing van deze schermen is dan ook wenselijk.
Afbeelding 19-4 geeft schermsoort 2 weer.



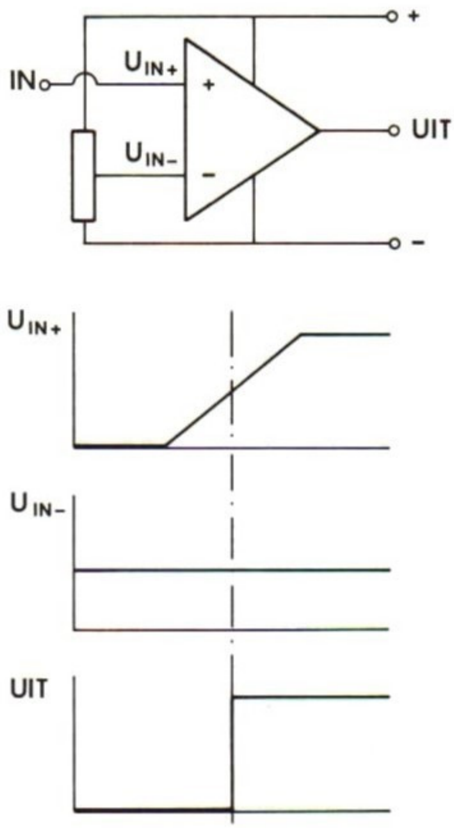
Afbeelding 19-4 De opbouw van scherm 2.

Een horizontale beeldlijn is opgebouwd uit 256 beeldpunten. Helemaal links ligt beeldpunt 0, terwijl beeldpunt 255 zich geheel aan de rechterzijde bevindt. Elk beeldpunt laat zich met de PSET-opdracht aansturen.
Een horizontale lijn op het hogeresolutiescherm is opgedeeld in groepjes van acht beeldpunten. Binnen zo'n groepje van acht mag slechts één enkele kleur voorkomen (deze beperking geldt niet voor punten langs verticale lijnen).
Hoewel deze wet de grafische mogelijkheden van de MSX beperkt, betekent het voor ons project een aanzienlijke vereenvoudiging. Door de beperking is het minder nuttig een optische lezer te ontwikkelen waarmee elk beeldpunt afzonderlijk kan worden uitgelezen als we die punten toch niet individueel kunnen beïnvloeden. We mogen op een minder fijne schaal werken. Dat sluit prachtig aan op de mogelijkheden van het andere grafische scherm: scherm 3 (zie afbeelding 19-5).



Afbeelding 19-5 De opbouw van scherm 3.

Scherm 3 wordt ook wel het multicolor-scherm genoemd, omdat de beperking van het hogeresolutiescherm hier niet aanwezig is. Elk beeldpunt kan elke mogelijke kleur aannemen. Een beeldpunt bestaat echter uit een vlakje van vier bij vier hogeresolutiepunten.
Hoewel de beeldpunten dus aanzienlijk groter zijn, is op een horizontale lijn het kleuroplossend vermogen tweemaal zo groot. Terwijl op het hogeresolutiescherm horizontale lijntjes van acht punten eenzelfde kleur moesten hebben, kunnen we op het multicolor-scherm lijntjes van vier beeldpunten een kleur toekennen.
De optische lezer, die we nodig hebben om beelden op papier om te zetten in elektrische signalen, moet in staat zijn de gemiddelde grijswaarde te meten van een vlakje dat ongeveer overeenkomt met de afmeting van een lageresolutiebeeldpunt: een vierkantje van ongeveer een halve bij een halve centimeter.
Ten aanzien van de keuze van de benodigde analoog/digitaal-omzetter gelden ook beperkende voorwaarden. Het eenvoudigst is een omzetter die slechts twee niveaus kent. Twee niveaus worden namelijk al gedetecteerd door de eenvoudige vergelijkerschakeling uit afbeelding 19-6.



Afbeelding 19-6 Een eenvoudige vergelijkerschakeling.

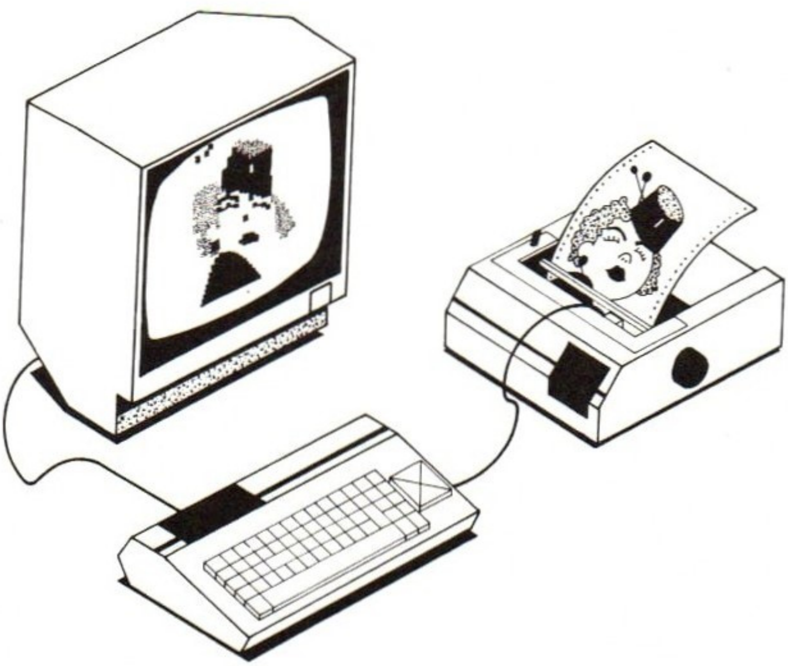
Als de ingangsspanning boven het met de potentiometer ingestelde niveau ligt, is de uitgang 'hoog'. Ligt de ingangsspanning eronder, dan is de uitgang 'laag'. Met deze schakeling kunnen we een beeld in een zwart/witplaatje omzetten. Maar dan ook letterlijk zwart/wit, dus zonder tussenliggende grijstinten. Hoewel met de geschetste opzet best aardige dingen te doen zijn, is de beperking wel erg groot. Er bestaat ook een ander uiterste.

Stel dat we kiezen voor een ADC met een acht-bits-uitgang. Een voor de hand liggende gedachte omdat de in de MSX toegepaste Z80-microprocessor met woorden van acht bits werkt. We kunnen dan 2 tot de macht 8 ofte wel 256 kleurtinten vastleggen. Maar hoe doen we dat op een computer met 'slechts' 16 kleurmogelijkheden? Als we een zwart/wit-TV aansluiten en we gaan ervan uit dat elke op de MSX beschikbare kleur een goed te onderscheiden grijswaarde oplevert, dan komen we al niet verder dan 15 grijswaarden (transparant telt niet mee). Bovendien willen we het gebruik van kleur niet opgeven, omdat het nu juist de charme aan de computer verleent.

De volgende tabel geeft de kleurmogelijkheden van MSX-systemen, iets anders gerangschikt dan in de meeste handboeken:

Kleur	*Nummer*	*Kleur*	*Nummer*
lichtrood	9	lichtgroen	3
rood	8	groen	2
donkerrood	6	donkergroen	12
cyaan	7	helderwit	15
lichtblauw	5	wit	14
donkerblauw	4	lichtgeel	11
paars	13	geel	10
transparant	0	zwart	1

De kleuren rood en groen zijn met drie helderheden vertegenwoordigd. Eigenlijk geven alleen deze kleurschakeringen een harmonische contrastcombinatie. Een ADC met drie niveaus lijkt voor een niet al te ingewikkeld beeldverwerkingssysteem dan ook aantrekkelijk. De consequentie van het toepassen van een ADC met drie niveaus is duidelijk. Het beeld verliest aan nuancering en wordt opgedeeld in drie helderheidsgraden. Alle grijswaarden binnen een bepaald helderheidsgebied worden bij elkaar geveegd. In de fotografie staat dit verschijnsel bekend als toonscheiding. Door het indringende effect wordt toonscheiding veel gebruikt als kunstzinnige uiting. Ook in de reclamefotografie komen we toonscheiding vaak tegen. Dat met slechts drie 'tonen' toch aantrekkelijke resultaten te realiseren zijn, laten de afbeeldingen in dit boek zien.

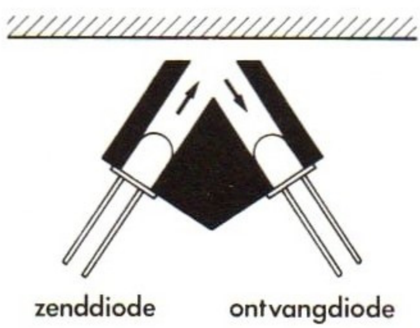


Afbeelding 19-7 Het beeldverwerkingssysteem.

Als laatste punt dient de beeldopnemer zich aan. We willen lijnen aftasten. We kunnen de beeldopnemer langs een liniaal bewegen en om de paar millimeter een helderheidsmeting doen. Vervolgens verschuiven we de liniaal iets naar onderen en tasten de volgende lijn af. Omslachtig en onpraktisch, nietwaar?
Vrijwel elke computerbezitter beschikt over een printer. U raadt het al: een printkop beschrijft een horizontale lijn en keert daarna terug naar het begin van de regel. Vervolgens wordt het papier een regel opgeschoven en een nieuwe lijn wordt afgewerkt. Als we op de bewegende printkop een lichtgevoelig opnemertje monteren en we de op te nemen afbeelding op het printerpapier plakken, krijgen we een prachtige beeld-scanner.
In de volgende hoofdstukken komen de opnemer, de koppeling met de computer, de analoog/digitaal-omzetter en de ombouw van de printer aan de orde. Dit laatste overigens zonder de printer ingrijpend te wijzigen. Afbeelding 19-7 geeft een indruk van het hele beeldverwerkingssysteem.

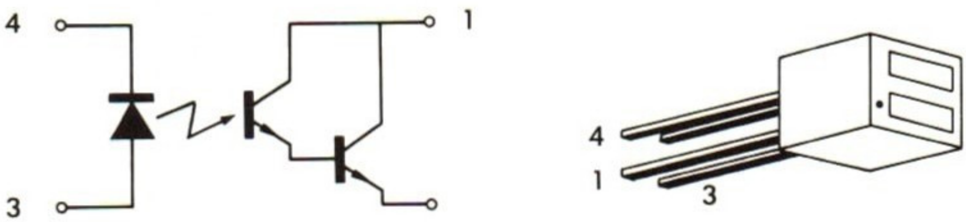
20 De fotogevoelige opnemer

Zoals we in het vorige hoofdstuk hebben afgeleid, moet het lichtgevoelige opneemelement werkzaam zijn op een oppervlakte van ongeveer een kwart vierkante centimeter. Er zijn uiteraard meer eisen te verzinnen. De opnemer moet compact en licht zijn. We kunnen van een printer niet eisen dat hij een grote print vol elektronicacomponenten meevoert.
Globaal gesproken heeft een printer één seconde nodig om een volledige lijn af te werken. Op een horizontale lijn van screen 3 liggen 64 beeldpunten. De opnemer moet dus in staat zijn per seconde 64 lichtveranderingen te registreren. Laten we er gemakshalve van uitgaan dat de opnemer een frequentiebereik van honderd hertz moet hebben. Hierdoor komt een lichtgevoelige weerstand (LDR) niet voor onze toepassing in aanmerking. Een fototransistor of -diode is echter bij uitstek geschikt.
De opnemer moet zo ongevoelig mogelijk zijn voor omgevingslicht. Een fotodiode in het infraroodgebied zou al heel wat problemen met daglichtafscherming oplossen. Het spreekt welhaast vanzelf dat we als lichtbron in dit geval een infrarood-LED gebruiken. Afbeelding 20-1 laat een opstelling zien voor een zend- en ontvangdiode. De ontvangdiode zet het door de afbeelding gereflecteerde licht om in een elektrische stroom.



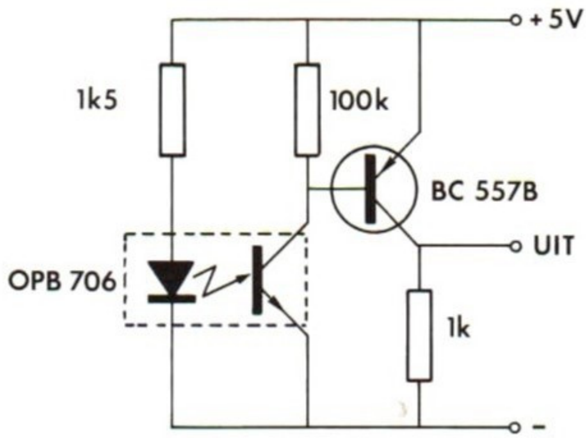
Afbeelding 20-1 De zend- en ontvangdiode, beide werkzaam in het infraroodgebied.

Zend- en ontvangdioden voor infrarood licht zijn ruimschoots verkrijgbaar. Met enige handigheid kan het afgebeelde opnemertje zelf worden vervaardigd.
Nog fraaier is de toepassing van een kant en klaar element, zoals de OPB-706, de CNY-70 of één van de vele varianten. Deze opnemer is bedoeld om diffuus reflecterende oppervlakken af te tasten en voldoet dan ook bij uitstek aan het door ons gestelde doel.



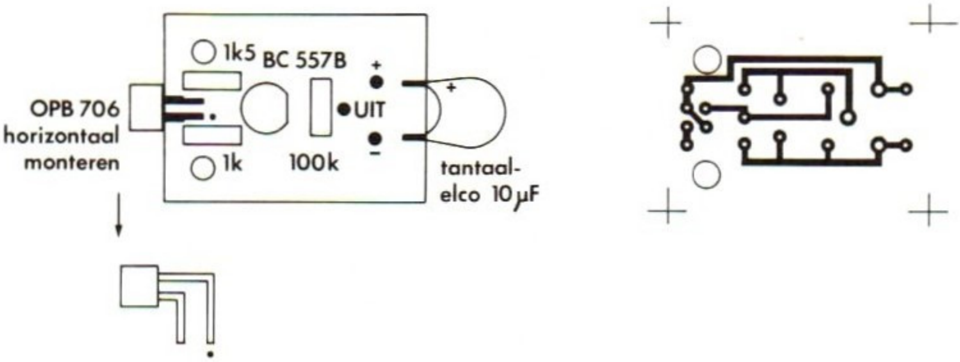
Afbeelding 20-2 De OPB-706: een combinatie van een infrarood zenddiode en een infrarood ontvangdiode.

Het gebruik van een fotodiode of een fototransistor heeft wel tot gevolg dat we flink wat signaalversterking in moeten bouwen. En dat zal dan met een minimum aan onderdelen moeten gebeuren, omdat we maar een klein printje in de printer kunnen bouwen. Het schema uit afbeelding 20-3 helpt ons al een aardig eind op weg.



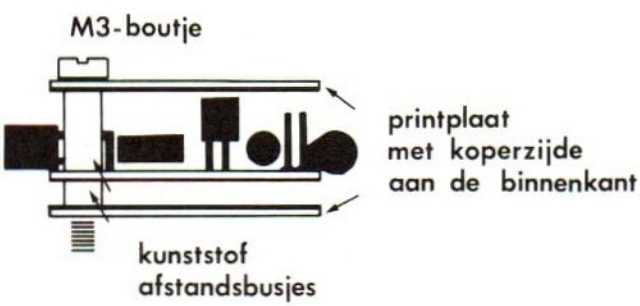
Afbeelding 20-3 Het schema van de lichtopnemer.

De opnemer, een PNP-transistor, drie weerstanden en een tantaal-elco. Dat is, behalve het printje, alles wat we nodig hebben voor de ontvanger. Voor het zendgedeelte komt daar nog een weerstand bij.
Het printje heeft ook al niet veel om het lijf. De lezer die in staat is zelf printen te vervaardigen, zal met dit miniprintje geen enkel probleem hebben. De sporen kunnen op het printmateriaal worden getekend met een etsbestendige pen. Ook kunnen etsbestendige afwrijfsymbolen worden gebruikt; die zijn tegenwoordig in vrijwel elke elektronicazaak te krijgen. Het kan ook zonder geëtste print. Een stukje gaatjesprint geeft voldoende basis voor het stabiel verbinden van de onderdelen.
Door de aftaster (sensor) een plaatsje aan de rand van de print te geven, kunnen we voor elke printer een optimale stand realiseren. Zo kan de sensor ge-



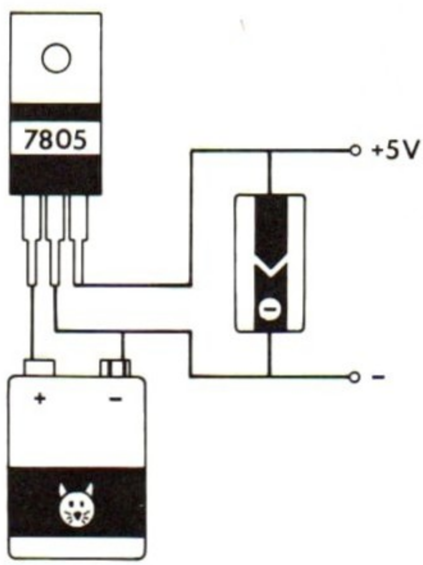
Afbeelding 20-4 De print-layout en de componentenopstelling van de lichtopnemer.

woon op de print worden gesoldeerd, waarbij de lengte van de poten aan de omstandigheden kan worden aangepast. Laat ze niet langer dan strikt noodzakelijk, omdat het draadeinde tussen de ontvangdiode en de versterkeringang brom kan oppikken. In de meeste gevallen is het handig als de sensor evenwijdig aan het printoppervlak wordt gemonteerd.
De print is zó klein, dat de bevestiging op de printerslede geen onoverkomelijke moeilijkheden hoeft op te leveren. In hoofdstuk 23 komen we hierop terug.
Mocht het nodig zijn dan kan de hoeveelheid LED-licht aan de omstandigheden worden aangepast door de weerstand van 1,5 kΩ te veranderen. Een lagere waarde dan 270 Ω is echter niet toegestaan. Dit in verband met de maximaal toegestane LED-stroom.
De taantaal-elco zorgt voor een extra ontkoppeling van de voedingslijnen. Om ruimte te besparen, worden de aansluitsnoeren rechtstreeks in de daarvoor bestemde printgaatjes gesoldeerd. De tantaal-elco wordt hierbij misbruikt als trekontlasting. Een plakbandje om de aansluitdraden en de condensator voorkomt dat de aansluitingen door herhaald verbuigen, afbreken. Neem voor de aansluitdraden dun en soepel geïsoleerd montagedraad van voldoende lengte. Vlecht de draden in elkaar. Eventueel kan dun tweeaderig afgeschermd snoer (stereo-kabel) worden toegepast. In dat geval wordt een ader gebruikt



Afbeelding 20-5 De afscherming van de lichtopnemer.

voor het signaal en een voor de voedingsspanning. De afscherming dient als massaverbinding.
Vanwege de hoge signaalversterking is het raadzaam om het printje af te schermen. Dit kan op een zeer eenvoudige manier. Twee miniprintplaatjes, aan elke zijde één, maken er een sandwich van (zie afbeelding 20-5). Ze voorkomen dat omgevingsbrom wordt opgepikt. Door middel van een draadje komt de massa-aansluiting met de print tot stand.
We gebruiken de printplaatjes tevens voor een ander doel. De afstand van de sensor tot het papier waarop de over te nemen afbeelding staat, dient redelijk constant te zijn. Immers, de hoeveelheid gereflecteerd licht dat de ontvanger bereikt, wordt beïnvloed door de afstand tot het reflecterende oppervlak.
De laatste schets geeft aan dat de afschermende printplaatjes, in het geval dat de sensor evenwijdig aan de print is gemonteerd, voorkomt dat de afstand tot het papier varieert.
Twee M3-boutjes met de bijbehorende kunststof afstandsbusjes vervolmaken het geheel tot een zeer stabiele en compacte leeskop. De boutjes geven bovendien goede mogelijkheden voor de montage in de printer.
Hoewel de uiteindelijke schakeling rechtstreeks vanuit de computer wordt gevoed, is het raadzaam in de testfase een aparte 5 V-voeding te gebruiken, om elk risico te vermijden. Een 5 V-spanning kan heel eenvoudig van een 9 V-batterijtje worden afgeleid. Een spanningsregelaar van het type 7805 en een elco van 470 μF zorgen voor een goed bruikbare spanningsbron (zie afbeelding 20-6).
Vóór we de ADC onder de loep nemen, testen we de leeskop door hem op een zwart gedeelte van een krant te zetten. De uitgangsspanning dient in dat geval ongeveer één volt te bedragen. Plaatsen we de leeskop op een onbedrukt ge-



Afbeelding 20-6 Een alternatieve voedingsbron.

deelte van de krant, dan staat er vier tot vijf volt op de uitgang. Een grijs gedeelte van een foto zal een waarde van ongeveer twee volt opleveren. Deze waarden zijn slechts een indicatie en hangen af van de afstand tussen de sensor en het papier, de papiersoort en de gebruikte inktsoort. Zo absorberen sommige soorten viltstiftinkt vrijwel geen infrarood licht. Wat voor ons oog een hoog contrast oplevert, hoeft dus niet per se hetzelfde effect op te leveren in het infraroodgebied van het spectrum. De versterking kan eventueel nog wat worden aangepast door de waarde van de terugkoppelweerstand (10 MΩ) te veranderen.

21 De koppeling met de computer

Alvorens in dit hoofdstuk de analoog-naar-digitaal-omzetting uit te werken, zullen we de mogelijkheden eens op een rij zetten die de MSX-computer heeft om met de buitenwereld te communiceren. Dat zijn er nogal wat. De standaarduitvoering heeft maar liefst zes aansluitingen waar we een signaal in de computer kunnen voeren:

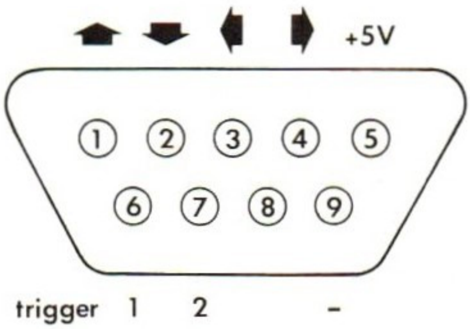
- Twee joystick-ingangen.
- Een ingang voor de cassetterecorder.
- Een gleuf waarin allerlei uitbreidingen zoals Quickdisk en ROM-packages aansluiting vinden.
- Een Centronics-aansluiting waarop een printer kan worden aangesloten.
- Een interface-aansluiting waarop vrijwel alles aanwezig is om het computergebeuren van buitenaf te regelen.

De laatste aansluiting geeft de meeste toepassingsmogelijkheden. Met enige kennis van de interne adreshuishouding, wat decodeerwerk en twee 'bus-IC's' behoort de bouw van een universele interface tot de mogelijkheden. Maar voor we naar het handboek en de soldeerbout grijpen, is enige bezinning op zijn plaats.
Laten we ons nog even goed realiseren wat ons uitgangspunt is: Een spanning omzetten naar drie discrete niveaus, en aan de hand van het geldende niveau een beeldpunt kleuren. We hebben gezien dat alleen de kleuren rood en groen elk in drie schakeringen voorkomen. Nog zonder te kiezen voor een bepaalde koppeling kan een programma er als volgt uitzien:

```
10  REM beeldpunt
20  SCREEN 2:COLOR 15,4,1
30  OPEN "grp:" FOR OUTPUT AS #1
40  PRINT #1,"[redacted]";
50  PRINT #1,"Positie x,y";
60  INPUT X,Y
70  INPUT NIVEAU
80  IF NIVEAU=1 THEN C=9:REM donkerrood
90  IF NIVEAU=2 THEN C=8:REM rood
100 IF NIVEAU=3 THEN C=6:REM lichtrood
110 PSET (X,Y),C
120 GOTO 120
```

Het behoeft geen betoog dat het zinloos is om het hier afgedrukte programma ook daadwerkelijk in te typen. De regels 70 t/m 100 laten echter duidelijk zien dat het binnenhalen van de gewenste gegevens heel goed kan via één van de joystick-ingangen. Daar hebben we immers de beschikking over meer dan voldoende onafhankelijke kanalen, namelijk de kanalen die de richtingen omhoog, omlaag, naar links en naar rechts representeren. Bovendien kunnen ze

diagonale bewegingen detecteren en er zijn ook nog twee 'vuurknop'aansluitingen (zie afbeelding 21-1).



Afbeelding 21-1 De joystick-ingang van een MSX-computer.

Herschreven voor de joystick-ingang nummer 1, ziet het programma er als volgt uit:

```
10  REM joystick
20  SCREEN 2:COLOR 15,4,1
30  OPEN "grp:" FOR OUTPUT AS #1
40  PRINT #1,"████████████████";
50  PRINT #1,"Positie x,y";
60  INPUT X,Y
70  A=STICK(1)
80  IF A=3 THEN C=9:REM donkerrood
90  IF A=5 THEN C=8:REM rood
100 IF A=7 THEN C=6:REM lichtrood
110 PSET (X,Y),C
120 GOTO 120
```

Regel 40: 16 keer [GRAPH][P]

De analoog-naar-digitaal-omzetter die het werk voor ons moet gaan doen, wordt aangesloten op de joystick-ingangen rechts, omlaag en links.
Om dit te illustreren laat het volgende programma een beeldpunt langs horizontale lijnen over het scherm lopen. Hierbij wordt de kleur met de cursorbesturing op het toetsenbord gekozen.

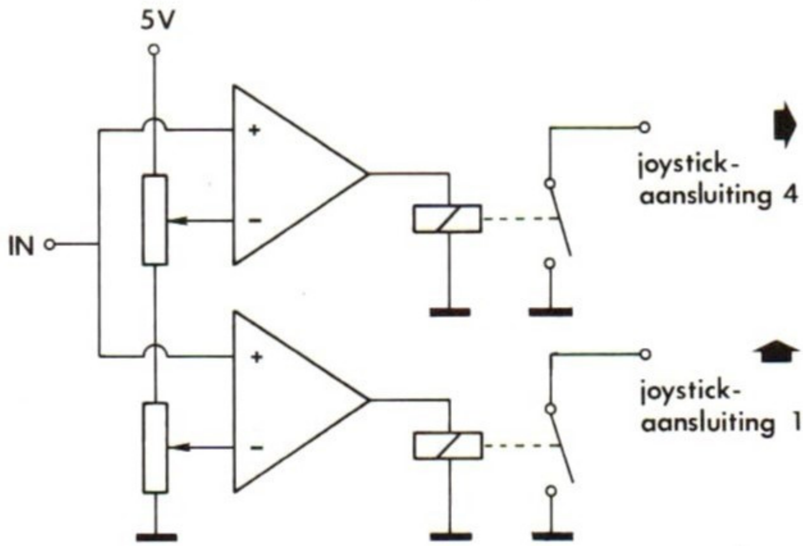
```
10  REM pixels in rood
20  SCREEN 3:COLOR 1,1,1:CLS
30  FOR Y=0 TO 192 STEP 4
40    FOR X=0 TO 255 STEP 4
50      A=STICK(0)
60      IF A=3 THEN C=9
70      IF A=5 THEN C=8
80      IF A=7 THEN C=6
90      IF A<>3 AND A<>5 AND A<>7 THEN 110
100     PSET (X,Y),C
```

```
110   NEXT X
120 NEXT Y
130 GOTO 130
```

Zodra het programma is gestart, wordt het beeldscherm zwart. Druk op één van de cursortoetsen en de beeldstip trekt een lijn in de overeenkomstige kleur. Regel 90 zorgt ervoor dat bij het indrukken van een verkeerde toets (de toets met de pijl omhoog of een combinatie van toetsen) de laatst geselecteerde kleur gehandhaafd blijft. Het lijkt onwaarschijnlijk, maar het zojuist ingetypte programma is op software-gebied vrijwel het enige dat nodig is om afbeeldingen gedigitaliseerd in het videogeheugen op te slaan.

De analoog-naar-digitaal-omzetter

Eigenlijk verdient de schakeling die we hier gaan beschrijven de naam ADC niet helemaal.
Omdat we slechts drie lichtniveaus willen detecteren, hebben we genoeg aan twee vergelijkingsschakelingen (comparators). Afbeelding 21-2 laat het schema van de comparator zien.



Afbeelding 21-2 De comparator.

Aan één van de ingangen van de beide comparators wordt het door de leeskop afgegeven signaal aangeboden, terwijl op de andere ingang een referentiespanning staat. Door de referentiespanningen voor beide comparators apart instelbaar te maken, kunnen we twee niveaus vastleggen waarbij de uitgangsspanning omklapt. Ligt het signaal onder de twee niveaus, dan zijn beide comparatoruitgangen laag. Tussen de twee niveaus in is één ervan hoog terwijl de

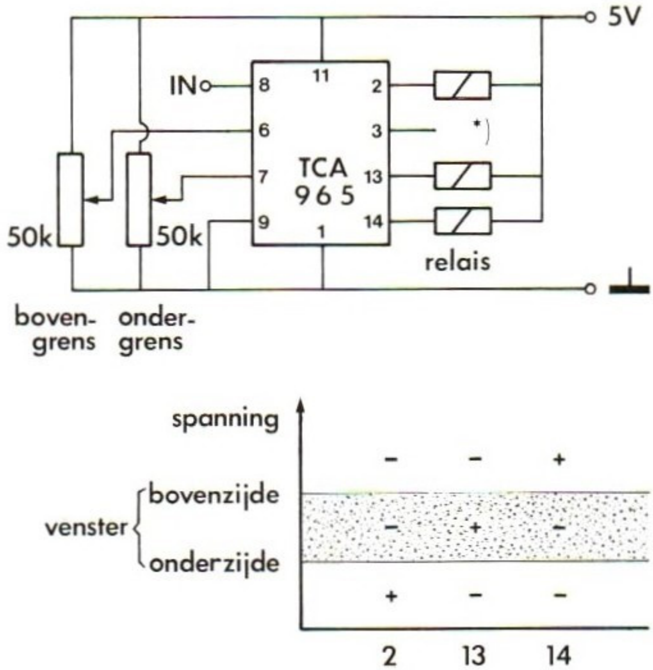
andere laag is. Bij een ingangssignaal dat zich boven het tweede niveau bevindt, zijn beide uitgangen hoog.
Twee relais, die direct door de schakeling worden gestuurd, nemen de functie van de joystick-schakelaars over.
Deze schakeling heeft een aardige eigenschap: we kunnen volstaan met het gebruik van slechts twee aansluitingen van de joystick-ingang. Bijvoorbeeld pen 1 en pen 4, die overeenkomen met omhoog en rechts.
Het schema uit afbeelding 21-2 laat zien dat, wanneer geen relais aanspreekt, STICK(1) de waarde 0 oplevert. Als het onderste relais aanspreekt, heeft STICK(1) de waarde 1 en als beide relais aanspreken (pen 1 en pen 4 aan massa) krijgt STICK(1) de waarde 2 (schuin rechts omhoog). In een programma vertalen we deze waarden naar kleuren.

```
IF STICK(1)=Ø THEN COLOR 6  (donkerrood)
IF STICK(1)=1 THEN COLOR 8  (rood)
IF STICK(1)=2 THEN COLOR 9  (lichtrood)
```

In groen ziet dat er vergelijkbaar uit:

```
IF STICK(1)=Ø THEN COLOR 12 (donkergroen)
IF STICK(1)=1 THEN COLOR  2 (groen)
IF STICK(1)=2 THEN COLOR  3 (lichtgroen)
```

De besproken schakeling is eigenlijk een vensterindicator. Een vensterindicator bekijkt of een spanning onder, in, of boven een vooraf ingesteld venster ligt. Deze functie komt in de meet-en-regeltechniek veel voor als bewaker van processen. In zo'n proces kan het voorkomen dat een temperatuur zich tussen twee uitersten moet bevinden. Of dat van een voorwerp de afmetingen tussen zekere grenzen moeten liggen. Een vensterindicator kan dan 'goedgekeurd' (binnen de grenzen) of 'afgekeurd' (buiten de grenzen) aangeven.
Vanwege het veelvuldige gebruik van vensterindicatoren is het niet verwonderlijk dat er een IC bestaat dat de functie van vensterindicator zelfstandig kan vervullen. De TCA 965 (Siemens) is zo'n IC. Afbeelding 21-3 laat één van de manieren zien waarop dit IC te gebruiken is.
Een bijkomend voordeel van dit IC is de ingebouwde hysteresis. Kleine variaties van het ingangssignaal in de buurt van een schakelpunt resulteren hierdoor niet in een storend geratel van de aangesloten relais. We passen relais toe, ondanks het feit dat een directe koppeling zoveel eenvoudiger is. De reden hiervoor kunt u raden. Een directe koppeling met het binnenste van de computer is ongewenst omdat het, bij een eventuele storing, niet erg plezierig is als een gedeelte van de computer wordt opgeblazen.
Om het hier beschreven project ook voor de computer- hardware-leek geschikt te laten zijn, zullen we elk risico bij dit project dan ook zorgvuldig mijden en gebruik maken van reed-relais. Reed-relais zijn klein, passen in een IC-voet en kunnen rechtstreeks vanuit het venster-IC worden gestuurd. Ze voldoen bovendien aan de door ons gespecificeerde maximale schakelfrequentie van

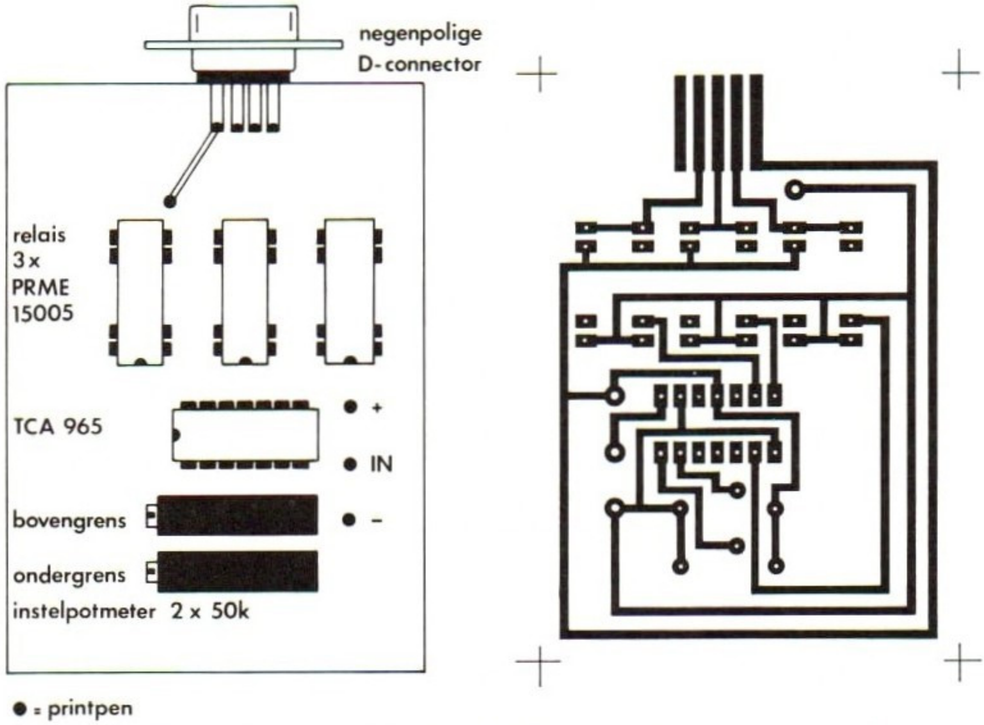


Afbeelding 21-3 Het gebruik van de vensterindicator TCA 965.
**) Uitgang 3 van de TCA 965 geeft aan als het ingangssignaal zich buiten het venster bevindt en wordt door ons niet gebruikt.*

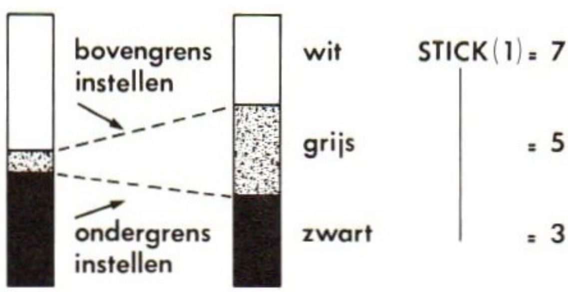
honderd hertz. De print-layout laat zien dat de joystick-plug op de print wordt bevestigd. Voor dit doel is een chassisdeel uitstekend geschikt. Chassisdelen in de gewenste uitvoering zijn algemeen verkrijgbaar in de serie D-connectoren. Bij sommige MSX-computers ligt de joystick-ingang te diep in de kast om een chassisdeel te kunnen gebruiken. Voor die computers is een stekker aan een kabeltje de oplossing.
Omdat we het gebruik van dubbelzijdige printen willen vermijden, gebruiken we de pennen 2, 3 en 4 van de joystick-aansluitplug. Alleen voor de massa-aansluiting is dan een draadbrugje nodig.
Zoals de print aangeeft, betrekken we de 5 V-voedingsspanning rechtstreeks uit de computer. In totaal trekt de schakeling iets meer dan 20 mA. Dit levert voor de voeding van de MSX-computer geen enkel probleem op. Omdat we op deze wijze toch met (in dit geval minder kwetsbare) computer-elektronica spelen, is het zaak om ook deze schakeling eerst met een externe 5 V-voeding uit te proberen. De eerder genoemde batterij met een 5 V-regelaar kan ook nu weer voortreffelijke diensten bewijzen.

De afregeling

Als de schakeling is aangesloten op een 5 V-voeding, is de afregeling ervan kinderlijk eenvoudig. We sluiten de sensor aan en zetten de potentiometers in de middenstand. Als we nu met een wit papiertje de leeskop tot op ongeveer



twee millimeter naderen, schakelen de reed-relais: we horen ze tikken. Bij een bepaalde afstand tussen de sensor en het papier vibreren de reed-relais: het venster-IC detecteert een grenswaarde en twijfelt tussen het wel of niet aanspreken van het relais.
We zoeken wat foto's in een krant op. Misschien ligt de krant er nog van de vorige test, toen de werking van de sensor werd beoordeeld.
We zetten de sensor op een wit gedeelte van de krant en meten de uitgangsspanning. Vervolgens plaatsen we de sensor op een gemiddeld grijs gedeelte en meten ook nu weer de uitgangsspanning. Tot slot meten we de uitgangsspanning als de sensor op een zwart vlak staat.
Stel de potentiometers dusdanig in, dat de reed-relais aanspreken zoals in afbeelding 21-5 grafisch staat weergegeven.

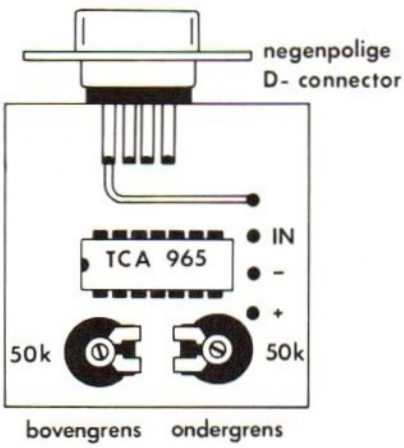


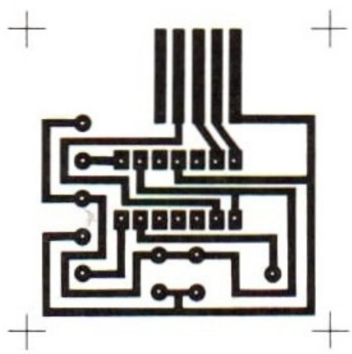
Afbeelding 21-5 De instelling van de potentiometers.

Deze afregeling voldoet voorlopig. Eventueel kunnen we in een later stadium de potentiometers nog een beetje bijstellen, afhankelijk van de aard van de op te nemen afbeelding.

22 Een schakeling voor gevorderde elektronici

Omdat u dit hoofdstuk leest, nemen we aan dat u een vaste hand van solderen heeft en bovendien de nodige kennis van elektronica. Op basis hiervan kunnen we de ADC nog aanzienlijk vereenvoudigen. De reed-relais, die in het vorige hoofdstuk een plaatsje op de ADC-print hebben gekregen, zitten daar uitsluitend om risico's voor de beginnende elektronicus te vermijden. De uitgangen van het besproken venster-IC liggen in de actieve toestand aan massa. Ze verrichten dus dezelfde functie als een joystick-schakelaar. Het printje komt er aantrekkelijk eenvoudig uit te zien als de relais vervallen. Vanwege de betrouwbaarheid van het IC, de zeer geringe belasting ervan voor de computer en het feit dat de joystick-ingang in het ergste geval – als werkelijk alles verkeerd gaat – alleen maar aan de voedingsspanning van de computer zelf komt te liggen, lopen we geen echt risico als de vensterindicator rechtstreeks met de joystick-ingang wordt verbonden.





Afbeelding 22-1 De vereenvoudigde comparatorschakeling: print-layout en componentenopstelling.

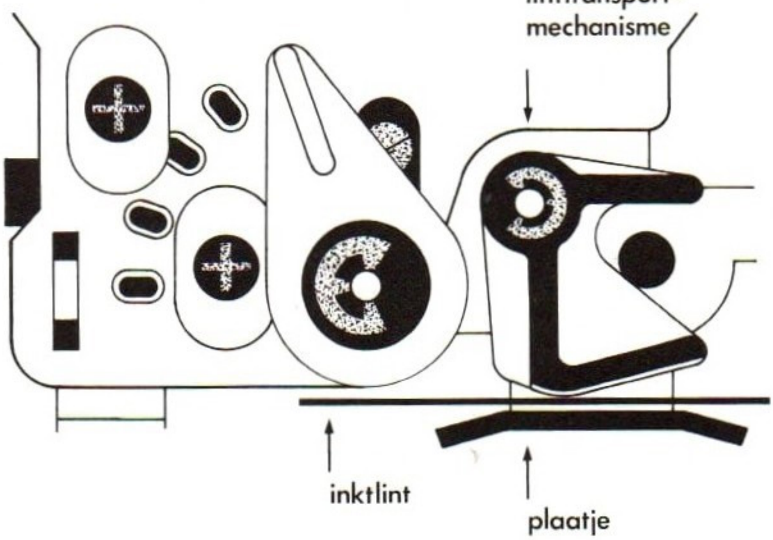
Het is zelfs de vraag of voor deze eenvoudige schakeling wel een speciaal geprepareerd printje nodig is. In een kwartiertje zetten we de hele schakeling op een stukje gaatjesprint in elkaar.
Als de ruimte in de printer die voor de experimenten gebruikt gaat worden dat toelaat, is de gedachte om de opnemer en de vensterindicator op een printje te combineren erg aantrekkelijk. Een soepel snoertje met stekker, zoals dat aan een joystick zit, geeft het geheel een professioneel uiterlijk. Het uiterlijk kan nog verder verfraaid worden door het geheel in een geschikte kunststoffen behuizing te bouwen.

23 Printeraanpassing

Het is niet de bedoeling dat de printer ingrijpend wordt omgebouwd. Als u in de luxe omstandigheid verkeert in het bezit te zijn van een printer die zijn beste tijd heeft gehad, kan de aanpassing natuurlijk definitief worden doorgevoerd. We zullen niet gaan knutselen aan de printer zelf en verrichten zelfs geen handelingen die een eventueel nog lopende garantieperiode in gevaar kunnen brengen.
In normale gevallen is de printer op de computer aangesloten via de printerkabel. De computer stuurt gecodeerde tekens over de kabel naar de printer. Zodra de printerbuffer een volle regel heeft opgenomen, stuurt de printer het 'busy'-signaal naar de computer, om aan te geven dat de computer moet wachten met het verzenden van nieuwe tekens. De printkop gaat lopen en de regel wordt afgedrukt. In de tijd dat de printer de regel afdrukt, staat de computer op non-actief. Bij sommige printers kan de buffer meer dan één regel opslaan en kan de computer tijdens het printen gewoon doorwerken. We beschouwen hier een eenvoudige matrixprinter met een buffer van slechts één regel. Tijdens de beweging van de printkop willen we de lichtintensiteit langs die regel meten en opslaan. Dit is echter nogal moeilijk als de printer tijdens deze slag de computer uitschakelt. We kunnen het 'busy' uitschakelen door de desbetreffende draad in één van de aansluitpluggen van de printerkabel los te nemen. In dat geval moeten we toch nog iets doen om de juiste spanning op de losgekoppelde aansluitpunten te krijgen. Het zonder meer doorknippen van een kabel-ader geeft niet het gewenste resultaat. We blijven bij ons uitgangspunt en laten ook de kabel intact. Sterker nog, we sluiten de printerkabel helemaal niet aan!
Vrijwel iedere printer kan een zelftest uitvoeren. Tijdens een zelftest worden alle tekens die de printer kent, afgedrukt. Na elke regel wordt bovendien het papier een regel verder gedraaid; een proces dat zich steeds herhaalt. De door ons gewenste scanbeweging wordt tijdens de zelftest dus netjes uitgevoerd, zonder dat de computer geblokkeerd raakt. We kunnen de over te nemen afbeelding op het kettingpapier plakken en het zelftestmechanisme in werking stellen. Op een kleinigheid na... het is niet de bedoeling dat de met zorg gekozen afbeelding vol met tekens wordt bedrukt. We kunnen dit op twee manieren voorkomen. We kunnen de elektrische verbinding naar de printkop losnemen; bij vrijwel alle printers is dat een tamelijk eenvoudige zaak. Mocht dat echter bij sommige printers minder eenvoudig blijken te zijn, dan verwijderen we eenvoudigweg het inktlint, zodat de printer geen zichtbare sporen op de afbeelding achterlaat. Wordt dit laatste door de printerfabrikant afgeraden, dan zit er niets anders op dan tussen het lint en het papier waarop de te kopiëren afbeelding zich bevindt een papiertje te schuiven, zodat de printtekens hierop afgedrukt worden in plaats van op de over te nemen afbeelding.

De mechanische aanpassing

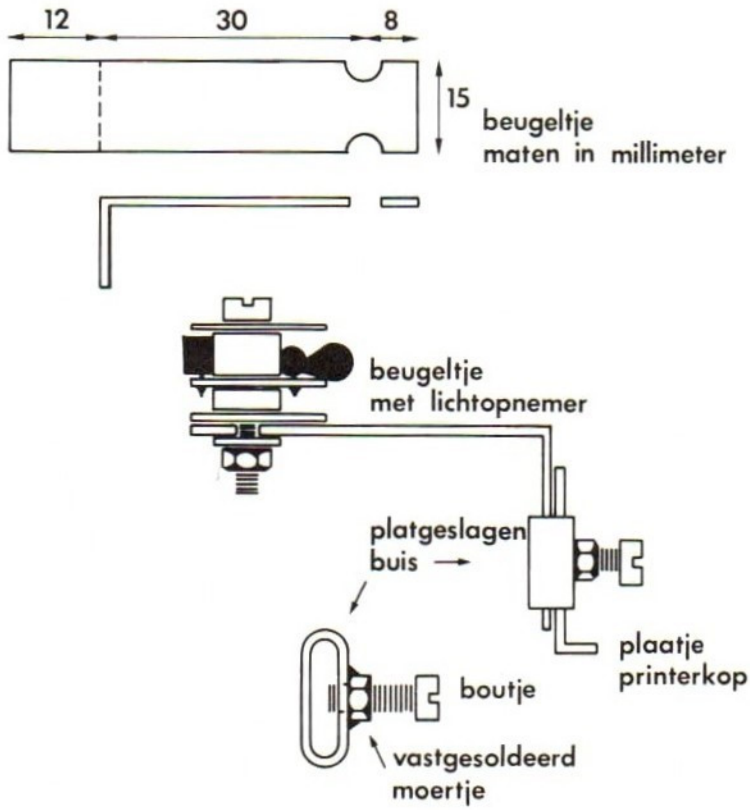
Het is te veel gevraagd om in dit boek voor alle printers te beschrijven hoe de leeskop het best op het bewegende deel van de printer kan worden bevestigd. Met enige inventiviteit is er in de meeste gevallen wel een oplossing te vinden. Voor een populaire printer geven we hier een oplossing. Voor andere printers kan een kleine aanpassing tot het gewenste effect leiden. In de echt moeilijke gevallen kan de leeskop ook met dik dubbelzijdig plakband worden bevestigd. Ook een stukje plasticine (boetseerklei) kan voor de oplossing in nood zorgen.
De GP-100 van Seikosha is een robuuste matrixprinter die bij veel computerbezitters wordt aangetroffen. Afbeelding 23-1 laat zien dat het inktlint op de terugweg langs een haaks omgezet metalen plaatje wordt geleid. Er tegenover zit een palletje dat ervoor zorgt dat het lint tijdens het printen naar één kant wordt meegenomen.



Afbeelding 23-1 De printerkop van de GP-100 van Seikosha.

Het plaatje vormt een stevig geheel met de rest van de constructie en is er als het ware voor gemaakt om de leeskop aan te bevestigen. Voor dit doel hebben we twee hulpstukjes nodig. In de eerste plaats een metalen beugeltje, dat bijvoorbeeld uit een strookje aluminium wordt gemaakt. Eventueel kan een kant en klaar beugeltje uit de ijzerwinkel of hobbyzaak op maat worden gezaagd. De breedte ervan is niet erg belangrijk, mits die niet méér dan anderhalve centimeter bedraagt.
Afbeelding 23-2 laat zien hoe de leeskop op het beugeltje wordt bevestigd met behulp van de M3-boutjes op de leeskop. Het tweede hulpstukje is een metalen cilindertje met een hoogte van één centimeter (iets minder mag ook) en een diameter van ongeveer 15 millimeter. Zeer geschikt hiervoor is een stukje

koperen waterleidingbuis. We drukken het stukje buis plat en boren er vervolgens een gaatje in. Met een M3-tapje kunnen we schroefdraad in het gat aanbrengen. We moeten in dit geval een gaatje boren met een diameter van 2,5 millimeter. Mocht u geen tapje hebben, dan kunt u ook een M3-moertje op de platgeslagen buis solderen. Het gat waarvoor het moertje komt te zitten, moet dan een diameter hebben van 3 of 3,5 millimeter.



Afbeelding 23-2 De bevestiging van de lichtopnemer met het beugeltje en het hulpstuk om het beugeltje met de lichtopnemer op de printerkop te bevestigen.

Let er goed op dat de leeskop vrijloopt van papiergeleiders en andere obstakels. Doordat de printslede iets verend is opgehangen, rust de leeskop met een lichte druk tegen het papier. De afstand tot het papier is hierdoor zeer stabiel.
Sommige printers hebben geen zelftestmogelijkheid die van buitenaf te starten is. Bij de GP-100 kunnen we deze mogelijkheid zelf installeren door de punten 15 en 35 van de aansluitplug (aan de achterzijde van de printer) met elkaar te verbinden. Een miniatuurschakelaartje dat in een plug wordt gemonteerd, verricht deze klus op eenvoudige wijze. De rasechte werktuigbouwkundige kan het schakelaartje ook in de printer monteren.
We zijn er bijna, en zijn nu zover dat we kunnen proefdraaien. Hiervoor brengen we in het al besproken programma nog enkele wijzigingen aan.

```
10  REM ongesynchroniseerd
20  SCREEN 3:COLOR 1,1,1:CLS
30  FOR Y=0 TO 192 STEP 4
40    FOR X=0 TO 255 STEP 4
50      K=STICK(1)
60      IF K=0 THEN C=9
70      IF K=5 THEN C=8
80      IF K=6 THEN C=6
90      PSET(X,Y),C
100   NEXT X
110   A$=INPUT$(1)
120 NEXT Y
130 GOTO 130
```

Breng een afbeelding in de printer door deze op het kettingpapier te plakken. Plak de bovenkant met een strookje plakband vast. Neem hiervoor niet-reflecterend plakband, zoals bijvoorbeeld Scotch-tape. Zet ook de zijkanten van de afbeelding met een stukje plakband vast. Let erop dat de leeskop geen loszittende randjes papier mee kan nemen. Gebruik daarom indien mogelijk een kopie op A4-formaat. Kies bij voorkeur een afbeelding met hoge contrasten.
Zet de printer op zelftest en start het programma (RUN [Return] of F5). Telkens als de printer is teruggekeerd naar de uitgangspositie, moet u de spatiebalk even indrukken. Op het scherm verschijnt het eerste beeld. Misschien nog wat onherkenbaar omdat de synchronisatie met de hand niet erg optimaal is. Daar gaan we nu wat aan doen.

De synchronisatie

Het synchronisatieprobleem lossen we op met een microschakelaar die is ingedrukt als de leeskop zich in de uitgangspositie (links) bevindt. Een geschikt plaatsje voor die schakelaar is altijd wel te vinden. Bij de GP-100 is de steun waarover de linker inktlintcassette wordt geschoven, uitstekend geschikt. Zelfs het gat zit op de goede plaats. Met een afstandsbusje wordt de microschakelaar zó gemonteerd dat hij aan het begin van de slag door de leeskop wordt ingedrukt. Bij sommige printers kan de schakelaar met het al eerder genoemde dubbelzijdige plakband op een geschikte plaats worden aangebracht. Neem een microschakelaar met een lichte schakeldruk en let erop dat het schakelarmpje lang genoeg is. We gebruiken het maakcontact. Eén van de schakelcontacten wordt door een lange draad met de massa van de leeskop verbonden. De draad moet de volledige printslag kunnen volgen. Het andere contact van de schakelaar verbinden we met pen 6 van de joystick-plug. Deze pen is de triggeringang en bevindt zich aan de componentenzijde van de vensterindicatorprint. Met STRIG(1) kan worden bepaald of de schakelaar al dan niet is ingedrukt.
In het programma nemen we de automatische synchronisatie op in een paar aanvullende regels.

```
10  REM gesynchroniseerd
20  SCREEN3:COLOR 1,1,1:CLS
30  IF STRIG(1) THEN 30 ELSE 40
40  FOR Y=0 TO 192 STEP 4
50    FOR X=0 TO 255 STEP 4
60      K=STICK(1)
70      IF K=3 THEN C=9
80      IF K=5 THEN C=8
90      IF K=7 THEN C=6
100     PSET(X,Y),C
110   NEXT X
120   IF STRIG(1) THEN 130 ELSE 120
130   IF STRIG(1) THEN 130 ELSE 140
140 NEXT Y
150 GOTO 150
```

Als de 256 punten langs de horizontale as in stappen van vier pixels (regel 50) zijn afgetast (regel 110), zorgt regel 120 ervoor dat het programma pas doorgaat als de schakelaar is ingedrukt. Regel 130 zorgt ervoor dat de volgende lijn op het scherm verschijnt als de leeskop vrijkomt van de schakelaar. Het begin van elke beeldlijn op het scherm komt door de synchronisatie steeds weer nauwkeurig overeen met de kantlijn van het papier. In het volgende hoofdstuk zullen we nog enkele verfijningen in het programma aanbrengen.

24 Het werken met de digitizer

In de voorafgaande hoofdstukken zijn de hardware-zaken aan de orde geweest. In dit hoofdstuk gaan we verder in op de software-kant van het gebeuren.

Het zal u waarschijnlijk al zijn opgevallen dat de synchronisatie af en toe een steekje laat vallen. Toch lijkt het programma waterdicht. De computer maakt geen fouten. In dit geval doet hij zijn werk zelfs te goed. Door de hoge frequentie waarmee de eindschakelaar wordt uitgelezen (met de STRIG(1) opdracht), kan het gebeuren dat er al weer een meting wordt gedaan als de contacten van de schakelaar nog nadenderen. De computer interpreteert dat alsof de leeskop de uitgangspositie alweer heeft verlaten, terwijl die nog maar net gearriveerd is. Het beeldpunt begint dan ook alvast over het scherm te lopen, terwijl de leeskop nog rustig in de hoek staat. Voor dit probleem zijn meerdere oplossingen te verzinnen: we kunnen een condensator over de schakelaar plaatsen of we kunnen een vertraging in het programma aanbrengen:

```
125 FOR Q=1 TO 100:NEXT Q
```

Het eenvoudigst is niet het maakcontact, maar het verbreekcontact van de microschakelaar te gebruiken en de regels 30, 120 en 130 hierop aan te passen.

```
30  IF STRIG(1) THEN 40 ELSE 30
120 IF STRIG(1) THEN 120 ELSE 130
130 IF STRIG(1) THEN 140 ELSE 130
```

Het contact-dender-probleem is nu verlegd naar de start van de leescyclus. De tijdverschillen die hier optreden, zijn echter niet waarneembaar. Echter, bij minder goede schakelaars verloopt ook het verbreken van het contact niet helemaal betrouwbaar. Om het zekere voor het onzekere te nemen, bouwen we toch maar een wachtlus in en maken gewoon gebruik van het maakcontact.

```
10  REM gesynchroniseerd met wachtlus
20  SCREEN 3:COLOR 1,1,1:CLS
30  IF NOT STRIG(1) THEN 40 ELSE 30
40  FOR Y=0 TO 192 STEP 4
50    FOR X=0 TO 255 STEP 4
60      K=STICK(1)
70      IF K=3 THEN C=9
80      IF K=5 THEN C=8
90      IF K=7 THEN C=6
100     PSET(X,Y),C
110   NEXT X
120   IF NOT STRIG(1) THEN 120
```

```
130    FOR Q=1 TO 100:NEXT Q
140    IF STRIG(1) THEN 140
150 NEXT Y
160 GOTO 160
```

Dan lijkt nu alles rond. Regel 160 zorgt ervoor dat het beeld netjes op het scherm blijft staan totdat we CONTROL-STOP indrukken of de computer uitzetten. Dus tot zolang genieten maar van het mooie plaatje... Maar daarvoor hebben we alle moeite niet gedaan. We willen op zijn minst het plaatje bewaren om het in een programma als versiering te gebruiken. De spelletjesfanaten zullen de opgenomen beelden willen gebruiken als decor voor de meest spannende avonturen. De combinatie van sprites en opgenomen achtergrondbeelden is natuurlijk om van te watertanden. Door sprites op de juiste plaatsen te laten verdwijnen achter een sprite die de achtergrondkleur heeft, kunnen we zelfs de suggestie opwekken dat het beeld ruimtelijk is. De zaak is duidelijk: het beeld moet bewaard blijven, in welke vorm dan ook.
Voor de bezitters van een 3,5 schijfeenheid is er een gemakkelijke oplossing voorhanden. Een kleine programma- uitbreiding zorgt ervoor dat het volledige videogeheugen op schijf wordt gezet.

```
10   REM gesaved op schijf
20   SCREEN 3:COLOR 1,1,1:CLS
30   IF STRIG(1) THEN 40 ELSE 30
40   FOR Y=0 TO 192 STEP 4
50     FOR X=0 TO 255 STEP 4
60       K=STICK(1)
70       IF K=3 THEN C=9
80       IF K=5 THEN C=8
90       IF K=7 THEN C=6
100      PSET(X,Y),C
110    NEXT X
120    IF STRIG(1) THEN 120 ELSE 130
130    FOR Q=1 TO 100:NEXT Q
140    IF STRIG(1) THEN 150 ELSE 140
150 NEXT Y
160 BSAVE "BEELD1",0,16383,S
```

Een programma om de beelden terug op het scherm te zetten, is al even eenvoudig.

```
10 REM recover
20 BLOAD "BEELD1",S
30 GOTO 30
```

Let erop dat de naam die we voor het beeld kiezen (in dit geval 'BEELD1') slechts éénmaal op een diskettekant mag voorkomen. Voor elk beeld dat moet worden opgeslagen, moet u een andere naam verzinnen.
De beschreven opslagmethode is eenvoudig en geeft de volledige vrijheid het

beeld vanuit een programma op te roepen. Het laden van het beeld vraagt enige seconden, maar de beeldvorming zelf geschiedt zeer snel.
Er bestaat ook een manier om de beelden op een cassetteband op te slaan waarbij eventuele beeldbewerking achteraf zeer gemakkelijk is. De mogelijkheden hiertoe worden in volgende hoofdstukken onder de loep genomen.
Eerst poetsen we het beeld nog wat op. Het zou erg toevallig zijn als de lijnsnelheid op het scherm precies overeen zou komen met de snelheid waarmee de leeskop over het papier beweegt. In de meeste gevallen blijft de leeskop wat achter. Door de hogere schrijfsnelheid op het beeldscherm wordt het beeld in horizontale richting uitgerekt. We laten de computer in de pas lopen door het programma wat af te remmen. We doen dat door het inbouwen van een welbekende FOR-NEXT-wachtlus, direct nadat een punt is ingekleurd. Als de printer aan het einde van de regel is gekomen, zal de computer – vanwege de ingebouwde vertraging – nog niet aan het einde van het scherm zijn. De gegevens die de leeskop vanaf dat ogenblik doorgeeft, zijn echter niet meer relevant: de informatie die de leeskop tijdens de terugslag opneemt, hoort niet bij het op te nemen beeld. Behalve het vertragen van de beeldvorming maken we daarom ook de horizontale lijnen wat korter, door de X-coördinaat niet van 0 tot 255 te laten lopen, maar van 50 tot 200.

```
10  REM snelheidscorrectie
20  SCREEN 3:COLOR 1,1,1:CLS
30  IF STRIG(1) THEN 40 ELSE 30
40  FOR Y=0 TO 192 STEP 4
50    FOR X=50 TO 200 STEP 4
60      K=STICK(1)
70      IF K=3 THEN C=12
80      IF K=5 THEN C=2
90      IF K=7 THEN C=3
100     PSET(X,Y),C
110     FOR Q=1 TO 12:NEXT Q
120   NEXT X
130   IF STRIG(1) THEN 130 ELSE 140
140   FOR Q=1 TO 100:NEXT Q
150   IF STRIG(1) THEN 160 ELSE 150
160 NEXT Y
170 BSAVE  "BEELD1",0,16383,S
```

Hoe bevallen de nieuwe kleurtjes in de regels 70 tot en met 90?
De wachtlus in regel 110 en de coördinatenaanpassing in regel 50 kunnen per printer wat verschillen. U kunt ze naar wens aanpassen. Het beeld kan er ook opzettelijk mee worden vervormd. Op het onderwerp beeldbewerking komen we in een volgend hoofdstuk uitgebreid terug.

25 Beeldopslag

In het vorige hoofdstuk hebben we gezien hoe het gedigitaliseerde beeld vanuit het videogeheugen in één keer op een 3,5 inch-schijf kan worden gezet. Nu heeft niet iedereen een 3,5 inch-schijfeenheid. De cassetterecorder is nog steeds een veel gebruikt opslagmedium, maar ook de 'echte' diskdrive ligt inmiddels binnen het budget van menig computergebruiker.
We zoeken een methode waarbij de beeldpunten goed toegankelijk zijn voor verdere bewerking. In dit hoofdstuk gaan we daarom de beeldopslag drastisch wijzigen.
Van elk beeldpunt leggen we de X-coördinaat, de Y-coördinaat en de kleur vast. Door het slim aan te pakken kan dat op een zeer elegante manier. Het scherm waarop we werken (screen 3) heeft op de X-as (horizontaal) 64 posities. Op de Y-as (verticaal) treffen we 48 lijnen aan. Elke combinatie van X en Y wijst een beeldpunt aan. Elk beeldpunt heeft een eigen kleur. In de voorafgaande programma's werd de kleur aangeduid met een C. We blijven dit consequent doen, maar brengen een kleine uitbreiding aan. Behalve de kleuraanduiding worden ook de coördinaten, dus de plaats op het beeldscherm opgenomen.

```
C(X,Y)
```

De kleuren van de beeldpunten op het hele beeldscherm liggen vast met deze variabele. Zo zijn de kleuren van alle punten op regel 7 vastgelegd met C(X,7). Het derde beeldpunt op die regel heeft de kleur C(3,7). Als alle beeldpunten op één regel moeten worden afgebeeld, kan dat door X van 0 tot 63 (in totaal 64 beeldpunten) te laten lopen.

```
10 REM regel
20 FOR X=0 TO 63
30   PSET(X,7),C(X,7)
40 NEXT X
```

Om alle punten op het beeldscherm in te kleuren, dient bovendien de Y-coördinaat het traject van 0 tot 47 (in totaal 48 regels) te doorlopen.

```
10 REM scherm
20 FOR Y=0 TO 47
30   FOR X=0 TO 63
40     PSET(X,Y),C(X,Y)
50   NEXT X
60 NEXT Y
```

In het inleesprogramma dat in het vorige hoofdstuk tot stand is gekomen, gaan we de variabele C vervangen door C(X,Y). We mogen dat echter niet zonder meer doen. Arrays (in dit geval alle variabelen C(X,Y) bij elkaar) moeten van tevoren worden aangekondigd. De computer wil graag weten hoeveel geheugenruimte er voor arrays moet worden gereserveerd, omdat ze een zeer grote omvang kunnen aannemen. In ons geval gaat het om een array met 64 × 48=3072 elementen. De DIM-opdracht reserveert geheugenruimte voor de array.

```
DIM C(63,47)
```

N.B.: De kleur van het punt in de linker bovenhoek wordt door C(0,0) bepaald.
De oplettende lezer zal opmerken dat het beeldopnameprogramma niet de volle breedte van het scherm benut. We zouden dus kunnen volstaan met het niet tot het einde van de regel door laten lopen van de X-coördinaat. Een MSX-computer met 64K vrij geheugen heeft echter voldoende ruimte om alle beeldpunten op te bergen. En, in een later stadium gaan we het hele scherm bij beeldbewerking betrekken.
Het langzaam groeiende programma wordt geschikt gemaakt voor de toepassing van een array waarin de kleuren van de beeldpunten komen te staan:

```
10  REM beeldpunten inlezen
20  SCREEN 3:COLOR 1,1,1:CLS:KEY OFF
22  DIM C(63,47)
30  IF STRIG(1) THEN 40 ELSE 30
40  FOR Y=0 TO 47
50    FOR X=12 TO 50
60      K=STICK(1)
70      IF K=3 THEN C(X,Y)=12
80      IF K=5 THEN C(X,Y)=2
90      IF K=7 THEN C(X,Y)=3
100     PSET(4*X,4*Y),C(X,Y)
110     FOR Q=1 TO 6:NEXT Q
120   NEXT X
130   IF STRIG(1) THEN 130 ELSE 140
140   FOR Q=1 TO 100:NEXT Q
150   IF STRIG(1) THEN 160 ELSE 150
160 NEXT Y
```

Let vooral op de regels 100 en 110. De STEP 4-opdracht is vervallen. De factor vier is nu in regel 100 terug te vinden. Omdat regel 100 enige rekentijd vraagt, is de wachtlus in regel 110 iets ingekort om het beeld en de printer weer op elkaar af te stemmen. Bovendien is de laatste regel, waarin het beeld op schijf werd vastgelegd, vervallen. In plaats daarvan bouwen we het beeld direct weer op door de toevoeging van een nieuw stukje programma:

```
170 CLS
180 FOR Y=0 TO 47
190   FOR X=0 TO 63
200     PSET(4*X,4*Y),C(X,Y)
210   NEXT X
220 NEXT Y
230 GOTO 230
```

Vergelijk regel 50 eens met regel 190. In regel 50 lezen we het beeld in. Om het beeld netjes in het midden van het scherm te plaatsen en geen punten in te kleuren met 'terugslaginformatie', loopt de telling langs de horizontale as van 12 tot 50. In regel 190 daarentegen plaatsen we het beeld weer op het scherm. Alle punten lager dan X=12 en hoger dan X=50 zijn tijdens het opnemen niet gekleurd en hebben daarom de kleur 0: transparant. Dat wil zeggen dat ze op het scherm verschijnen als zwart, de achtergrondkleur.
De beeldpunten worden één voor één weer in de juiste kleur op het scherm gezet. Voor het opslaan van de afzonderlijke beeldpunten kunnen we het zojuist toegevoegde programmadeel wijzigen.

```
170 OPEN "CAS:COPIE" FOR OUTPUT AS #1
180 FOR Y=0 TO 47
190   FOR X=0 TO 63
200     PRINT #1,C(X,Y)
210   NEXT X
220 NEXT Y
230 CLOSE
```

Let er weer op dat de X-coördinaat de gehele lijn van 0 tot en met 63 doorloopt. Regel 170 opent het bestand op de cassetteband dat bestemd is om er gegevens in op te bergen (FOR OUTPUT). Het is natuurlijk mogelijk om elk ander opslagmedium te gebruiken. Voor een 3,5 inch-schijf zou regel 170 er als volgt uitzien:

```
170 OPEN "COPIE" FOR OUTPUT AS #1
```

Als de gegevens zijn weggeschreven, kan vrijwel hetzelfde programmadeel worden gebruikt om ze weer tevoorschijn te halen.

```
5  REM reproduktie
10 SCREEN3:COLOR 1,1,1:CLS
20 OPEN"CAS:COPIE" FOR INPUT AS #1
30 FOR Y=0 TO 47
40   FOR X=0 TO 63
50     INPUT #1,C(X,Y)
60     PSET(4*X,4*Y),C(X,Y)
70   NEXT X
80 NEXT Y
90 CLOSE
```

In regel 20 van het reproduktieprogramma wordt het bestand weer geopend, nu met de bedoeling er gegevens uit te halen (FOR INPUT). Het is erg onhandig als alle opgeslagen beelden met dezelfde naam (COPIE) op een cassettebandje staan. Op een 3,5 inch-schijf is dat zelfs niet eens toegestaan. Er bestaat een goede methode om een bestand te openen met een naam naar keuze. We doen dat door de naam apart op te vragen en de bestandsnaam samen te stellen uit een vast deel en een gedeelte dat steeds weer anders kan zijn.

```
10 INPUT "NAAM";N$
20 OPEN "CAS:"+N$+".DAT"
```

Hetzelfde geldt voor de 3,5 inch-schijf:

```
10 INPUT "NAAM";N$
20 OPEN N$+".DAT"
```

Probeer één van deze twee miniprogramma's maar eens en vul voor de naam COPIE1 in. Als alles goed gaat, komt er op het opslagmedium een leeg bestand te staan met de naam COPIE1.DAT.
Zoals bij de MSX-gebruiker bekend behoort te zijn, mag de naam uit niet meer dan acht tekens bestaan, terwijl het eerste teken een letter moet zijn. De drieletterige extensie kan weliswaar vrij worden gekozen, maar voor de herkenbaarheid is het goed om een bepaalde norm aan te houden. Zo wordt de toevoeging .DAT in het algemeen gebruikt voor DATA-bestanden. Als u de herkenbaarheid van de bestanden wilt verbeteren door .COP .DUP of iets dergelijks te gebruiken, is dat uiteraard geoorloofd, mits het voor uzelf maar herkenbare toevoegingen zijn. In dit boek zullen de toevoegingen .PIX en .SCR worden gebruikt voor beelden die als kleur-arrays (C(X,Y)) respectievelijk videogeheugen-dumps zijn opgeslagen.
Nu het opslaan van gegevens op tape of diskette geen problemen meer hoeft op te leveren, wordt het zo langzamerhand tijd om het beeldopnameprogramma in een goed hanteerbare vorm te gieten. Een aanzienlijk deel van het programma is al ingetypt, zodat volstaan kan worden met de toevoegingen. Met RENUM wordt de regelnummering weer op orde gebracht.

```
20  REM kopieer programma
30  KEY OFF:COLOR 1,10,10:CLS
40  PRINT"Breng een afbeelding in de printer"
50  PRINT"Druk op een toets en start"
60  PRINT"vervolgens de printerzelftest"
70  REM opnemen van het beeld
80  SCREEN 3:COLOR 1,1,1:CLS
90  DIM C(63,47)
100 IF STRIG(1) THEN 110 ELSE 100
110 FOR Y=0 TO 47
120   FOR X=12 TO 50
130     K=STICK(1)
140     IF K=3 THEN C(X,Y)=12
```

```
150       IF K=5 THEN C(X,Y)=2
160       IF K=7 THEN C(X,Y)=3
170       PSET(4*X,4*Y),C(X,Y)
180       FOR Q=1 TO 6:NEXT Q
190     NEXT X
200     IF STRIG(1) THEN 200 ELSE 210
210     FOR Q=1 TO 100:NEXT Q
220     IF STRIG(1) THEN 230 ELSE 220
230 NEXT Y
240 REM menu
250 SCREEN 0:COLOR 1,10,10:CLS
260 PRINT:PRINT:PRINT
270 PRINT" 1 Afbeelden op scherm"
280 PRINT" 2 Beeldpunten opslaan"
290 PRINT" 3 Dump op 3,5 inch-schijf"
300 PRINT" 4 Einde programma"
310 PRINT:PRINT:PRINT
320 PRINT"          Kies nummer";
330 A$=INPUT$(1):A=VAL(A$)
340 IF A<1 OR A>4 THEN 300
350 IF A=1 THEN 350
360 IF A=2 THEN 440
370 IF A=3 THEN 540
380 IF A=4 THEN CLS:KEY ON:END
390 REM afbeelden op scherm
400 SCREEN 3:COLOR 1,1,1:CLS
410 FOR Y=0 TO 47
420   FOR X=0 TO 63
430     PSET(4*X,4*Y),C(X,Y)
440   NEXT X
450 NEXT Y
460 A$=INPUT$(1)
470 IF A$=<>" " THEN 460
480 IF A=3 THEN 640 ELSE 250
490 REM beeldpunten opslaan
500 CLS:FILES
510 INPUT "Naam voor beeldpunten";N$
520 OPEN N$+".PIX" FOR OUTPUT AS #1
530 FOR Y=0 TO 47
540   FOR X=0 TO 63
550     PRINT #1,C(X,Y)
560   NEXT X
570 NEXT Y
580 CLOSE
590 GOTO 250
600 REM dump op 3,5 inch-schijf
610 CLS:FILES
620 INPUT "Geef naam voor scherm-dump op";N$
630 GOTO 390
640 BSAVE N$+".SCR",0,16383,S
650 GOTO 250
```

Het programma is in een aantal blokken verdeeld. De eerste regels geven instructie over het gebruik. Het tweede blok is het eigenlijke opnameprogramma. De te gebruiken kleuren worden vastgelegd in de regels 130 tot en met 150. Het is misschien een aardige BASIC-oefening om dit programma zodanig aan te passen dat de te gebruiken kleuren in het eerste programmablok kunnen worden ingevoerd. Na het opnemen meldt de computer zich met een menu waarin u kunt kiezen uit een aantal mogelijkheden. De laatste drie blokken zorgen ervoor dat wat u uit het menu kiest ook werkelijk gebeurt.
Let er goed op dat de vermenigvuldiging van X en Y met een factor 4 alleen maar noodzakelijk is als de beeldpunten op het scherm worden gezet. De factor ontbreekt bij het opslaan van de beeldpunten, omdat die daar overbodig is.
Nog een laatste opmerking. Om het beeld netjes in het midden van het scherm te krijgen, loopt de X-coördinaat tijdens de opname en de reconstructie van het beeld op het scherm niet van 0 tot en met 63. Als we de beeldpunten opslaan, doen we dat echter wél van 0 tot en met 63. De niet-gekleurde punten krijgen automatisch de kleurwaarde 0 (transparant). In een later stadium gaan we ook de nog niet door het beeld bezette zwarte marges vullen. Vandaar dat alle X-coördinaten worden opgeslagen.
Zo, dat zit erop. Hard- en software zijn behandeld en het is nu zaak om wat beelden te reproduceren. Wat dacht u van de portretten van uw familieleden, vrienden of kennissen? Gebruik contrastrijke beelden. Al eerder is vermeld dat sommige inktsoorten, zoals viltstiftinkt, slecht infrarood licht absorberen. Dat hoeft geen belemmering te zijn om ze toch door de computer te laten opnemen. U maakt dan namelijk eerst een fotokopie van de prent. Dat is in veel gevallen toch noodzakelijk, omdat papiervorm en -dikte printervriendelijk moeten zijn. Moderne kopieerapparaten bieden bovendien de mogelijkheid om afbeeldingen te vergroten of te verkleinen, waardoor de afmetingen naar wens kunnen worden aangepast. Een kopie op normaal A4-papier kan met een plakbandje aan boven- en zijkanten op het kettingpapier worden vastgezet.
In het volgende hoofdstuk wordt het pas echt leuk. We gaan onderzoeken in hoeverre het beeld verder bewerkt en veranderd kan worden.

26 Beeldbewerking

In de voorafgaande hoofdstukken hebben we de beelden opgebouwd in rood- of groenschakeringen.
Misschien heeft u zelf al met andere kleuren geëxperimenteerd. Omdat alle beeldpunten afzonderlijk kunnen worden opgeslagen, zijn we in staat om op elk beeldpunt bewerkingen los te laten. We kunnen dat echter ook op groepen van beeldpunten doen. Bijvoorbeeld op alle beeldpunten met het kleurnummer 6, of op alle beeldpunten waarvan de som van de X- en de Y-coördinaat gelijk is aan een bepaald getal. Het totale beeld kan worden verschoven door alle afzonderlijke beeldpunten te verplaatsen. Maar ook kunnen delen van het beeld worden verplaatst. Ook kunnen de punten met eenzelfde kleur als afzonderlijke beelden worden opgeslagen.
Plak een afbeelding met de bovenrand vast op het papier van de printer. Plaats aan de zijkanten ook een stukje plakband. We zijn er dan van verzekerd dat de afbeelding goed door het printerpapier wordt meegenomen. Als uw printer is uitgerust met een wrijvingsrol, in plaats van een pinfeed-mechanisme, dan zal het duidelijk zijn dat alle opmerkingen in dit boek over het opplakken van afbeeldingen niet voor u zijn bestemd.
Start het opnameprogramma en breng het beeld over op het scherm. Afhankelijk van de aard van het over te nemen beeld, zouden de drie kleurnuances ongeveer een gelijke oppervlakte moeten beslaan. Is dat niet het geval, dan is dit een mooie gelegenheid om de twee instelpotentiometertjes nog wat af te regelen. Een verdraaiing linksom laat het desbetreffende niveau zakken.
Kies uit het menu de optie waarbij alle beeldpunten worden opgeslagen en geef het bestand de naam 'COPIE'.
Ga na of op de band of de diskette nu het bestand COPIE.DAT voorkomt. Zo ja, dan is alles gelukt en voeren we na de opdracht NEW, die het geheugen ontruimt, het volgende programma in.

```
10   REM kleurverandering
20   DIM C(63,47)
30   SCREEN 0:COLOR 1,10,10:CLS:KEY OFF
40   FILES
50   INPUT "Naam ";N$
60   OPEN N$+".PIX"
70   SCREEN 3:COLOR 1,1,1:CLS
80   FOR Y=0 TO 47
90     FOR X=0 TO 63
100      INPUT #1,C(X,Y)
110      IF C(X,Y)=12 THEN C(X,Y)=4
120      IF C(X,Y)=3 THEN C(X,Y)=10
130      PSET(4*X,4*Y),C(X,Y)
140    NEXT X
```

```
150 NEXT Y
160 CLOSE
170 GOTO 170
```

Het resultaat is sprookjesachtig. De kleuren waarin het beeld opgenomen is, zijn naar keuze te wijzigen. In regel 110 en 120 kunnen we elke kleur toekennen aan twee van de drie oorspronkelijke kleuren. Door de toevoeging van nóg een regel kan ook de derde kleur naar wens worden veranderd.

```
122 IF C(X,Y)=2 THEN C(X,Y)=14
```

Het afgedrukte programma leent zich uitstekend voor variaties. We zullen het deel van het programma dat het beeld van het opslagmedium haalt apart zetten en er een blokje aan toevoegen waarmee het nieuwe beeld gesaved kan worden. Daarbinnen wordt ook een plaatsje ingeruimd voor de beeldbewerking zelf.

```
 10  REM beeldbewerking
 20  DIM C(63,47)
 30  SCREEN 0:COLOR 1,10,10:CLS:KEY OFF
 40  FILES
 50  INPUT "Naam ";N$
 60  OPEN N$+".PIX" FOR INPUT AS #1
 70  SCREEN 3:COLOR 1,1,1:CLS
 80  FOR Y=0 TO 47
 90    FOR X=0 TO 63
100      INPUT #1,C(X,Y)
130      PSET(4*X,4*Y),C(X,Y)
140  NEXT X
150  NEXT Y
160  CLOSE

200 GOSUB 1000
300 SCREEN 0:COLOR 1,10,10
310 FILES
320 INPUT "Naam ";N$
330 OPEN N$+".PIX" FOR OUTPUT AS #1
340 FOR Y=0 TO 47
350   FOR X=0 TO 63
360     PRINT #1,C(X,Y)
370   NEXT X
380 NEXT Y
390 CLOSE
```

De regels 110 en 120 zijn verdwenen. Regel 40 (alleen voor gebruikers van een 3,5 inch-schijfeenheid) zorgt ervoor dat alle bestanden die op de diskette staan, worden afgebeeld op het scherm. Een dubbel naamgebruik, met het daarbij behorende gevolg (oude bestand wordt overschreven), kan hierdoor worden vermeden. Let er bij het invoeren van een bestandsnaam wel op dat die zonder de toevoeging .DAT moet worden ingetypt.

Tussen het ophaalprogramma, dat loopt tot regel 160, en het wegschrijfprogramma, dat begint op regel 300, springt het programma naar een subroutine waarin de beeldbewerking plaatsvindt. Op deze manier kan iedere nog te verzinnen beeldbewerking als een op zichzelf staand programma worden ontwikkeld, mits dat begint met regelnummer 1000 en wordt afgesloten met een RETURN-opdracht.
De zojuist behandelde kleurverandering zetten we meteen maar om in een geschikte subroutine.

```
1000 REM kleurverandering
1000 FOR Y=0 TO 47
1010   FOR X=0 TO 63
1020     INPUT #1,C(X,Y)
1030     IF C(X,Y)=12 THEN C(X,Y)=4
1040     IF C(X,Y)=3 THEN C(X,Y)=10
1050     PSET(4*X,4*Y),C(X,Y)
1060   NEXT X
1070 NEXT Y
1080 IF STRIG(0) THEN RETURN ELSE 1080
```

De laatste regel oogt misschien wat merkwaardig, maar de regel zorgt ervoor dat het beeld, voordat het wordt opgeslagen, rustig kan worden bekeken. Pas na een druk op de spatiebalk wordt het save-proces in werking gesteld. Mocht u vooralsnog geen prijs stellen op het vastleggen van het bewerkte beeld, onderbreek dan het programma op de gebruikelijke manier met CTRL-STOP.
Als er belangstelling bestaat voor een negatief beeld, dan is dat zó voor elkaar: de donkerste en de lichtste kleur worden verwisseld.

```
1000 REM negatiefje
1000 FOR Y=0 TO 47
1010   FOR X=0 TO 63
1020     INPUT #1,C(X,Y)
1030     IF C(X,Y)=12 THEN C(X,Y)=2
1040     IF C(X,Y)=2 THEN C(X,Y)=12
1050     PSET(4*X,4*Y),C(X,Y)
1060   NEXT X
1070 NEXT Y
1080 IF STRIG(0) THEN RETURN ELSE 1080
```

Het volgende blok veroorzaakt bizarre effecten op het beeldscherm.

```
1000 REM random kleur
1010 C1=RND(1)*16:C2=RND(1)*16:C3=RND(1)*16
1020 FOR Y=0 TO 47
1030   FOR X=0 TO 63
1040     IF C(X,Y)=12 THEN C(X,Y)=C1
1050     IF C(X,Y)=3 THEN C(X,Y)=C2
1060     IF C(X,Y)=2 THEN C(X,Y)=C3
1070     PSET(4*X,4*Y),C(X,Y)
1080   NEXT X
1090 NEXT Y
1100 IF STRIG(0) THEN RETURN ELSE 1100
```

De hierna afgedrukte subroutine die aan het hoofdprogramma kan worden toegevoegd, maakt het mogelijk een kleur te isoleren. Een nieuw beeld, waarin slechts één van de kleurschakeringen is geselecteerd, wordt opgeslagen. Een mooie gelegenheid om ook de kleur even aan te passen.

```
1000 REM solokleur
1000 FOR Y=0 TO 47
1010   FOR X=0 TO 63
1020     INPUT #1,C(X,Y)
1030     IF C(X,Y)=12 THEN C(X,Y)=4
1040     IF C(X,Y)=3 THEN C(X,Y)=1
1050     IF C(X,Y)=2 THEN C(X,Y)=1
1060     PSET(4*X,4*Y),C(X,Y)
1070   NEXT X
1080 NEXT Y
1090 IF STRIG(0) THEN RETURN ELSE 1080
```

In dit geval is donkergroen vervangen door blauw. Groen en lichtgroen zijn verdwenen, omdat zwart tegen een zwarte achtergrond niet zichtbaar is.
Het subroutineblok laat zich op vele manieren samenstellen. Zo kan het tijdens het opnemen van een afbeelding gebeuren dat bepaalde punten in het gereproduceerde beeld ons toch niet zo goed bevallen. Bij een portret kan een lichtpuntje in de ogen wonderen doen. Het volgende blok geeft ons de mogelijkheid ieder afzonderlijk punt te beïnvloeden. We maken gebruik van de cursorbesturing op het toetsenbord, in combinatie met de spatiebalk.

```
1000 REM pixel verandering
1010 XO=0:YO=0:X=0:Y=0
1020 A=STICK(0)
1030 X=XO-(A=2)-(A=3)-(A=4)+(A=6)+(A=7)+(A=8)
1040 Y=YO+(A=1)+(A=2)+(A=8)-(A=4)-(A=5)-(A=6)
1050 PSET(4*X,4*Y),15
1060 FOR Q=1 TO 10:NEXT Q
1070 PSET(4*X,4*Y),1
1080 FOR Q=1 TO 10:NEXT Q
1090 IF STRIG(0) THEN GOSUB 1200
1100 IF X=XO AND Y=YO THEN 1020
1110 PSET(XO,YO),C(XO,YO)
1120 XO=X:YO=Y
1130 GOTO 1020

1200 REM subroutine kleurverandering
1210 A=STICK(0)
1220 IF A=1 THEN 300
1230 T=T+(A=7)-(A=3)
1240 IF T<1 THEN T=1
1250 IF T>15 THEN T=15
1260 C(X,Y)=T
1270 PSET(4*X,4*Y),C(X,Y)
1280 IF STRIG(0) THEN RETURN ELSE 1210
```

Nadat het eerste programmadeel (tot regel 170) het oorspronkelijke beeld op het scherm heeft gezet, verschijnt linksboven een knipperende cursor in beeld (regel 1050-1080).
Met de cursortoetsen kan het knipperlichtje naar elke plaats op het scherm worden bewogen. Zodra het naar een punt is gedirigeerd waarvan de kleur ons niet bevalt, drukken we op de spatiebalk (regel 1090).
De subroutine vanaf regel 1200 geeft de mogelijkheid om de kleur van het desbetreffende punt te wijzigen met behulp van de horizontale cursorbesturingstoetsen (regel 1230). Druk hierbij niet per ongeluk op de 'omhoog-toets'. Zodra de kleur naar tevredenheid is, brengt een tweede druk op de spatiebalk (regel 1280) ons terug in het hoofdprogramma. Na de correctie van het laatste beeldpunt, drukken we op de 'omhoog-toets' en het nieuwe beeld wordt opgeslagen. De cassetterecordergebruikers weten intussen welke veranderingen cq. toevoegingen nodig zijn om het programma voor hen geschikt te maken.
Met dit programma kunnen ook punten buiten het opgenomen beeld worden veranderd. Zo kunnen de linker- en rechtermarge worden ingekleurd. Echter, als deze gebieden overal dezelfde kleur moeten hebben, is dat sneller te realiseren door de achtergrondkleur in regel 70 aan te passen.
Voor de echte programmeur is dit programma een voorbeeld van minder fraai programmeerwerk. Het programma komt in een subroutine terecht, waar het niet meer uitkomt met een RETURN-opdracht, maar met een GOTO-opdracht. In dit geval verstoort het de loop van het programma echter niet, omdat na het opslaan van de gegevens het werk is gedaan.
Ter afsluiting van dit hoofdstuk bekijken we nog een techniek die in een later stadium van pas zal komen. Het programma is weer als subroutine uitgevoerd. Reclame-ontwerpers gebruiken de computer intensief. Eén van de methodes die zij gebruiken, is het omzetten van het beeld in horizontale lijnen met variërende dikte. Op het beeldscherm kunnen we dat enigszins nabootsen door de grafische karakters te gebruiken die met de toetsen u,i,o en p kunnen worden opgeroepen. Het beeld wordt dan niet in drie kleurschakeringen uitgevoerd maar de grootte van het blokje bepaalt de gemiddelde helderheid.

```
1000 REM lijndikte
1000 FOR Y=0 TO 47
1010   FOR X=0 TO 63
1020     INPUT #1,C(X,Y)
1030     IF C(X,Y)=12 THEN PRINT"▄ ";
1040     IF C(X,Y)=3 THEN PRINT"▬ ";
1050     IF C(X,Y)=2 THEN PRINT"▁ ";
1060     IF C(X,Y)=1 THEN PRINT"█ ";
1070   NEXT X
1080   PRINT" "
1090 NEXT Y
1100 IF STRIG(0) THEN RETURN ELSE 1100
```

Regel 1030: [GRAPH][o]
Regel 1040: [GRAPH][i]
Regel 1050: [GRAPH][u]
Regel 1060: [GRAPH][p]

In hoofdstuk 28, dat het afdrukken van bewerkte beelden behandelt, wordt een methode beschreven die hier erg veel op lijkt en die kan resulteren in een ongekend fraaie uitvoer op papier. We hebben nu een aantal bewerkingen mogelijk gemaakt die gebaseerd zijn op het manipuleren van kleuren. Het volgende hoofdstuk laat zien dat er met het verplaatsen van punten ook verrassende effecten te bereiken zijn.

27 Geometrische beeldbewerking

In het vorige hoofdstuk zijn enkele gereedschappen ontwikkeld, waarmee de kleuren van een opgenomen beeld kunnen worden gewijzigd. In dit hoofdstuk gaan we hetzelfde doen voor plaatsveranderingen van beeldpunten.
Het eerste deel van het programma uit het vorige hoofdstuk blijft ongewijzigd. Hiermee (regel 10 tot 170) wordt een opgeslagen beeld op het scherm gezet. De beeldbewerking verrichten we steeds in een aanvullend programmadeel, dat wordt toegevoegd aan het laadprogramma.
De afbeelding wordt tijdens de opname ervan gecentreerd op het beeldscherm geplaatst. Als de opname echter binnen andere programma's gaat worden gebruikt, is het helemaal niet gezegd dat die centrale plaats ook de juiste is. We beginnen dan ook met een verschuiving van het beeld in horizontale richting.



Afbeelding 27-1 Uw auteur opgebouwd uit pixels. Voor een goed beeld van de pixelplaatjes dient u het boek wat van u af te houden.

```
170 REM horizontale verschuiving
180 FOR Y=0 TO 47
190   FOR X=0 TO 63
200     XN=X-5
210     PSET(4*XN,4*Y),C(X,Y)
220   NEXT X
230 NEXT Y
240 GOTO 240
```

De verschuiving van het beeld langs de X-as blijkt niet zo moeilijk te zijn. In regel 200 wordt een nieuwe horizontale coördinaat XN berekend. In dit geval treedt er een verschuiving op van vijf beeldpunten in de negatieve richting,

naar links dus. In de PSET-opdracht wordt de kleur, die eigenlijk bij punt X behoort, afgebeeld op de nieuwe plaats XN. Het programma verricht deze operatie op alle beeldpunten van het scherm (regel 180 en 190). Als we in regel 200 XN=X+32 zetten, wordt het beeld aan de rechterzijde (gedeeltelijk) van het scherm geduwd.
Dezelfde methode is ook toe te passen voor een verticale verschuiving van de afbeelding. In dat geval berekenen we een nieuwe Y-coördinaat.

```
170 REM verticale verschuiving
180 FOR Y=0 TO 47
190   YN=Y-5
200   FOR X=0 TO 63
210     PSET(4*X,4*YN),C(X,Y)
220   NEXT X
230 NEXT Y
240 GOTO 240
```

Om te voorkomen dat de nieuwe Y-coördinaat bij elke X-waarde opnieuw moet worden berekend, zijn – in vergelijking tot het vorige programma – de regels 190 en 200 omgewisseld. Nu wordt in regel 190 de nieuwe Y-coördinaat (YN) berekend. Als dit programma als aanvulling op het hoofdprogramma uit het vorige hoofdstuk wordt uitgevoerd, blijkt dat er toch iets merkwaardigs aan de hand is. Het beeld wordt weliswaar over een afstand van vijf beeldpunten (regel 190) naar boven verschoven, maar de onderste vijf rijen van het oude beeld blijven staan. Dit euvel kunnen we gemakkelijk verhelpen door de FOR-NEXT-lus die de Y-waarden doorloopt, aan de nieuwe situatie aan te passen.

```
170 REM verticale verschuiving
180 FOR Y=0+5 TO 47+5
190   YN=Y-5
200   FOR X=0 TO 63
210     PSET(4*X,4*YN),C(X,Y)
220   NEXT X
230 NEXT Y
240 GOTO 240
```

Voor de duidelijkheid zijn de nieuwe Y-waarden in regel 180 niet uitgerekend. Door het op deze manier te schrijven, blijft de Y-verplaatsing herkenbaar. Het kan nog iets netter worden opgeschreven, zodat we de gewenste verschuiving maar op één plaats in hoeven te voeren.

```
170 REM verticale verschuiving
180 V=5
190 FOR Y=0+V TO 47+V
200   YN=Y-V
210   FOR X=0 TO 63
220     PSET(4*X,4*YN),C(X,Y)
230   NEXT X
240 NEXT Y
250 GOTO 250
```

Het beeldscherm wordt nooit helemaal door het opgenomen prentje gevuld. Toch is dat in veel gevallen wel gewenst. Natuurlijk kunnen we het beeld horizontaal wat uitrekken. De onderlinge verhoudingen gaan dan echter verloren en in veel gevallen is dat niet acceptabel. Omdat alle beeldinformatie beschikbaar is en het beeld bovendien op elke plaats kan worden neergezet, is het geschetste probleem voor een deel te omzeilen. We passen eenvoudigweg beeldverdubbeling toe. Het beeld wordt tweemaal naast elkaar geplaatst, en wel zodanig dat beide beelden de schermbreedte volledig benutten.
Vooral bij beelden met een horizon, maar ook bij geometrische motieven, kan dit alleraardigste effecten opleveren.



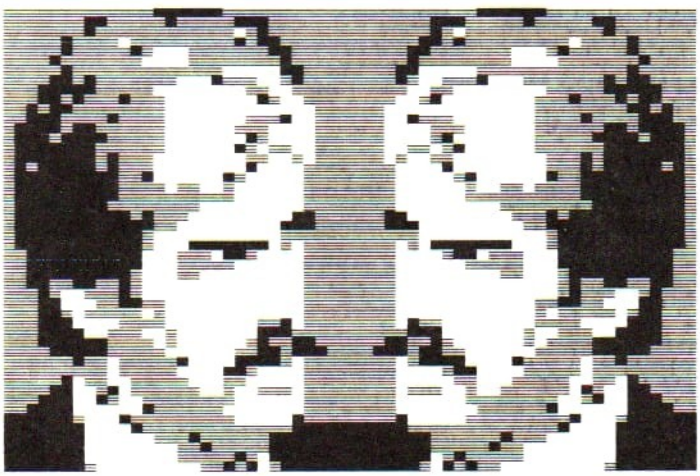
Afbeelding 27-2 Een portret van Kobus en een portret van Kobus met zijn tweelingbroer: beeldverdubbeling.

```
170 REM dubbelganger
180 FOR Y=0 TO 47
```

```
190    FOR X=0 TO 63
200      PSET(4*(X-12),4*Y),C(X,Y)
210      PSET(4*(X+31),4*Y),C(X+12,Y)
220    NEXT X
230 NEXT Y
240 GOTO 240
```

Als we het beeld kunnen verdubbelen, dan mag spiegelen ook geen grote problemen opleveren.

```
170 REM spiegel
180 FOR Y=0 TO 47
190    FOR X=0 TO 63
200      PSET(4*(X-12),4*Y),C(X,Y)
210      IF X>31 THEN 240
220      PSET(4*94-4*(X+31),4*YN),C(X+12,Y)
230    NEXT X
240 NEXT Y
250 GOTO 250
```

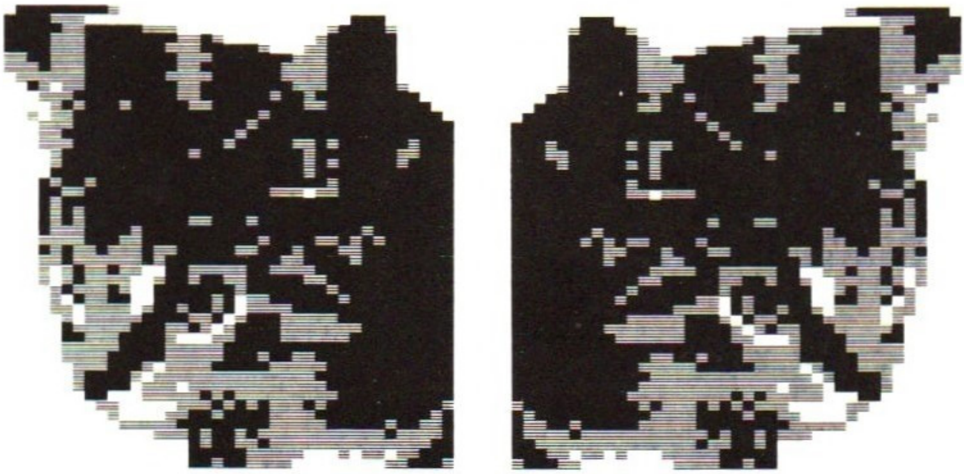


Afbeelding 27-3 De auteur kijkt zichzelf diep in de ogen: beeldspiegeling.

Regel 210 zorgt ervoor dat het gespiegelde rechterbeeld niet over het linkerbeeld wordt geschreven. In regel 220 laten we de Y-coördinaat van rechts naar links lopen. In het getal 94, dat voor die omkering zorgt, zit bovendien de rechter marge verrekend. De rechtermarge valt op deze wijze keurig buiten het beeld.
Voor het spiegelen van beelden komt er, zoals blijkt, wat eenvoudig hoofdrekenwerk om de hoek kijken. Veel simpeler is het omdraaien van het beeld. Dus links-rechts verwisseld.

```
170 REM links=rechts
180 FOR Y=0 TO 47
190    FOR X=0 TO 63
```

```
200       PSET(4*(62-X),4*Y),C(X,Y)
210     NEXT X
220 NEXT Y
230 GOTO 230
```



Afbeelding 27-4 Deze twee poezen willen niets meer met elkaar te maken hebben: beeldomkering.

Hiervoor hoeft de X-coördinaat slechts van rechts naar links, in plaats van van links naar rechts te lopen. We trekken de X-coördinaat telkens van de maximale X-waarde af. Voor spiegeling in de as die horizontaal halverwege het beeldscherm loopt, geldt iets dergelijks.

```
170 REM onder=boven
180 FOR Y=0 TO 47
190     FOR X=0 TO 63
200       PSET(4*X,4*(47-Y)),C(X,Y)
210     NEXT X
220 NEXT Y
230 GOTO 230
```

De beeldopbouw begint nu onderaan het scherm in plaats van bovenaan. Het op deze wijze gevormde beeld is een gespiegeld beeld, dat op z'n kop staat.
Om het beeld ongespiegeld ondersteboven op het scherm te zetten, kunnen we de programmadelen 'links=rechts' en 'onder=boven' combineren.

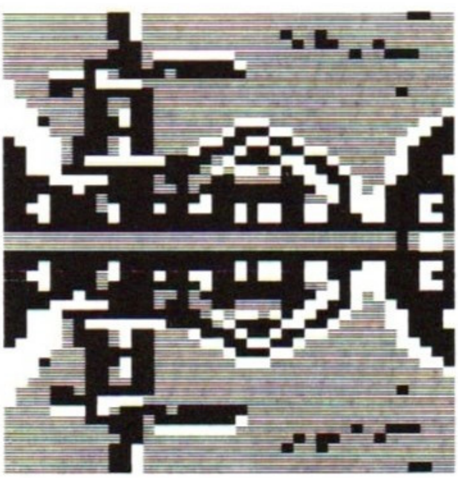
```
170 REM ondersteboven
180 FOR Y=0 TO 47
190     FOR X=0 TO 63
200       PSET(4*(62-)X,4*(47-Y)),C(X,Y)
210     NEXT X
220 NEXT Y
230 GOTO 230
```



Afbeelding 27-5 Een drukfout? Nee, het beeld staat op de kop door een dubbele beeldomkering.

Als we nu het TV-toestel of de monitor op z'n kop zetten hebben we het oude beeld terug.
Nog twee beeldgrapjes tot besluit.

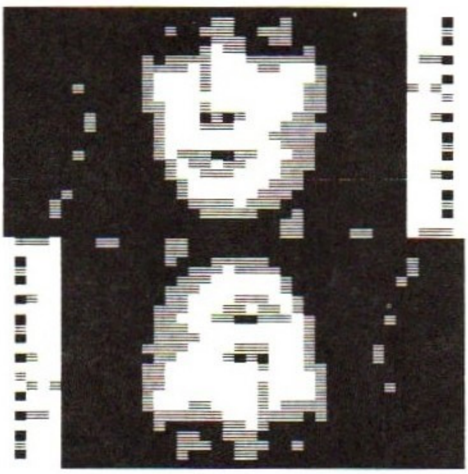
```
170 REM waterspiegel
180 FOR Y=0 TO 23
190   FOR X=0 TO 63
200     PSET(4*X,4*(47-Y)),C(X,Y)
210   NEXT X
220 NEXT Y
230 GOTO 230
```



Afbeelding 27-6 Noordhollands polderlandschap: waterspiegel.

Op het eerste gezicht is dit programmadeel identiek met het programmadeel 'onder=boven'. De tweede regel is echter verschillend. Door de Y-coördinaat maar tot de helft door te laten lopen, wordt de bovenste helft van de afbeelding in de onderste helft gespiegeld weergegeven. Een tafereeltje dat in ons land aan elke slootkant is te bewonderen.

```
170 REM speelkaart
180 FOR Y=0 TO 23
190   FOR X=0 TO 63
200     PSET(4*(62-X),4*(47-Y)),C(X,Y)
210   NEXT X
220 NEXT Y
230 GOTO 230
```



Afbeelding 27-7 Speelkaart: de dame.

Dit programma lijkt op de waterspiegel. In de onderste helft van het beeld wordt het gespiegelde beeld echter ook in de X-richting omgekeerd. Het prentje vertoont daardoor sterke gelijkenis met een speelkaart.
Het ziet er naar uit dat we de beeldopbouw aardig in de vingers beginnen te krijgen. We kunnen zelfs volslagen onzinnige operaties op de beeldpunten uitvoeren. Maar ook het coderen van een beeld behoort tot de mogelijkheden. Via een bepaalde sleutel kan het beeld worden vervormd. Alleen als de code bekend is, kan de reconstructie van het beeld plaatsvinden. Er zijn mogelijkheden te over, maar... genoeg gespeeld.
Het wordt tijd om de opgedane beeldbewerkingservaring in een gemakkelijk hanteerbaar programma te vatten. In een volgend hoofdstuk wordt de nodige aandacht aan zo'n meesterwerk besteed. Maar eerst bekijken we de mogelijkheden om een afdruk van het gedigitaliseerde beeld op papier te maken.

28 Uitvoer op de printer

MSX-computers zijn niet standaard voorzien van een screen dump-functie. Met zo'n functie kan met een bepaalde toetscombinatie de scherminhoud worden overgebracht op papier. In de literatuur zullen ongetwijfeld programma's te vinden zijn die in staat zijn deze functie te verrichten.
We zullen in dit boek geen screen dump-routine ontwikkelen. We maken gebruik van een methode die op alle matrix-printers een bevredigend resultaat zal geven, maar die ook op een margriet(daisy)wiel-printer bruikbaar is. Over de beeldkwaliteit van het laatste printertype moeten we ons niet al te veel illusies maken,omdat deze machines uitsluitend zijn bedoeld voor het weergeven van tekst. Toch is het beter dan niets. Als we van plan zijn een behoorlijk aantal prentjes op te nemen, kunnen we de printerafdrukken uitstekend gebruiken om een catalogus van die beelden samen te stellen.
Het omzetten van een pixelkleur naar een letter levert een redelijk resultaat op. De donkerste toon kunnen we op papier laten representeren door een X en de middentoon door x. Er ontstaat op deze wijze een zekere helderheidsvariatie.

```
IF C(X,Y)=12 THEN LPRINT "X";
IF C(X,Y)=2 THEN LPRINT "x";
IF C(X,Y)=3 THEN LPRINT "-";
IF C(X,Y)=1 THEN LPRINT " ";
```

Een streepje neemt de rol van de lichtste kleur op zich, terwijl spaties de achtergrond weergeven. De puntkomma zorgt ervoor dat de tekens direct achter elkaar worden afgedrukt.
Een printprogramma zou er dan zó uit kunnen zien:

```
 10 REM print
 20 DIM C(63,47)
 30 SCREEN 0:COLOR 1,10,10:CLS:KEY OFF
 40 FILES
 50 INPUT"Naam";N$
 60 OPEN N$+".PIX" FOR INPUT AS #1
 70 FOR Y=0 TO 47
 80   FOR X=0 TO 63
 90     INPUT #1,C(X,Y)
100     IF C(X,Y)=12 THEN LPRINT "X";
110     IF C(X,Y)=2 THEN LPRINT "x";
120     IF C(X,Y)=3 THEN LPRINT "-";
130     IF C(X,Y)=1 THEN LPRINT " ";
140   NEXT X
150   LPRINT
160 NEXT Y
170 CLOSE
```

In programmaregel 150 wordt het papier naar de volgende regel gedraaid. De uitvoer van dit programma ziet er een beetje vreemd uit. We hebben in het beeldopnameprogramma het beeld gecomprimeerd in horizontale richting. Dat moeten we nu eigenlijk weer compenseren. Door het aanbrengen van een kleine aanpassing in de regels 100 tot en met 130, bereiken we dat het beeld op papier er wat normaler uitziet.

```
100     IF C(X,Y)=12 THEN LPRINT "XX";
110     IF C(X,Y)=2 THEN LPRINT "xx";
120     IF C(X,Y)=3 THEN LPRINT "--";
130     IF C(X,Y)=1 THEN LPRINT "  ";
```

Alle tekens worden verdubbeld, waardoor er een resultaat ontstaat dat voor archiefdoeleinden bruikbaar is. Voor het op deze manier afdrukken van de afbeelding moet de printer wel 2×64=128 tekens op een papierregel kwijt kunnen. In het algemeen hebben de printers die bij homecomputers worden gebruikt slechts 80 kolommen. Als het plaatje niet de gehele breedte van het beeldscherm bestrijkt, kunnen we de X-coördinaat over slechts een gedeelte van de X-as laten lopen. We deden dat ook al in het beeldopnameprogramma.

```
80 FOR X=12 TO 50
```

Een beter alternatief wordt gevormd door, als dat tenminste mogelijk is, enkele tekens ('X', 'x' en '–') te gebruiken en de regelhoogte van de printer te halveren.

De matrixprinter

Vrijwel alle matrixprinters kennen een beperkte hoeveelheid grafische tekens. Door hiervan gebruik te maken, kan de kwaliteit van de afdruk iets worden verbeterd. De grafische tekens kunt u in de printerhandleiding opzoeken. Kies een teken met een donkere 'toon' uit. Een volledig zwart vlak voldoet nog het beste aan onze wens. Het teken moet wel de hele ruimte die er voor een teken beschikbaar is, opvullen, zodat er aan weerszijden van het afgedrukte teken geen witte randjes overblijven. Voor de middentint zoeken we een ander teken uit: een teken met een gemiddelde grijstint. De spatie dient als lichtste kleur. Voor de achtergrond kunt u een geschikt teken uitzoeken, maar u kunt ook hiervoor een spatie gebruiken. In de handleiding van de printer staat welke decimale codes bij de tekens horen. Is dat niet het geval, dan staat er in veel gevallen een hexadecimale code.
Voor MSX-printers zijn de tekens met de decimale codes 215 en 203 redelijk geschikt voor het weergeven van respectievelijk de donkerste 'toon' en de midden-'toon'. De spatie die de lichtste kleurschakering representeert, heeft het getal 255 als decimale code. Voor de achtergrond zou het teken met code 248 kunnen worden toegepast.
We zullen een ander voorbeeld bekijken aan de hand van een goedkope ther-

mische printer: de Brother HR-5. Als speciale grafische tekens zijn de tekens met de hexadecimale codes 8E en FE bij uitstek geschikt om als donkerste 'toon', respectievelijk midden-'toon' te dienen. De spatie heeft het getal 20 als hexadecimale code, terwijl de achtergrond opgevuld kan worden door tekens met de code 8F.
Het afdrukken van een teken met een decimale karaktercode gaat als volgt:

```
LPRINT CHR$(249);
```

De puntkomma zorgt ervoor dat de printer het volgende teken er direct achter plaatst. Voor het afdrukken van een teken met een hexadecimale karaktercode moeten we het voorvoegsel '&H' aan de code toevoegen.

```
LPRINT(&H8E);
```

Het printen op een MSX-printer gaat nu als volgt:
Breng het beeld vanaf het opslagmedium in het geheugen met behulp van het beeldweergaveprogramma, waarin de regels 100 tot en met 130 worden vervangen door:

```
100     IF C(X,Y)=12 THEN LPRINT CHR$(215);
110     IF C(X,Y)=2 THEN LPRINT CHR$(203);
120     IF C(X,Y)=3 THEN LPRINT CHR$(255);
130     IF C(X,Y)=1 THEN LPRINT CHR$(248);
```

Om de hoogte-breedte-verhouding van het beeld in de juiste verhouding op papier te krijgen, moet de printer op halve regelafstand worden gezet.
Voor een doorsnee-matrixprinter, moeten de regels 100 tot en met 130 er zó uit komen te zien:

```
100     IF C(X,Y)=12 THEN LPRINT CHR$(&H8E);
110     IF C(X,Y)=2 THEN LPRINT CHR$(&HFE);
120     IF C(X,Y)=3 THEN LPRINT CHR$(&H20);
130     IF C(X,Y)=1 THEN LPRINT CHR$(&H8F);
```

Ook hier moet worden geprint met halve regelafstand. Kijk in uw printerhandleiding hoe de regelafstand gehalveerd kan worden.
Een veel beter resultaat kan worden bereikt door zelf tekens te ontwerpen die de gewenste 'toon' hebben. Bovendien kunt u ze dan op elkaar laten aansluiten. Hiervoor dient de printer geschikt te zijn voor het afbeelden van bitpatronen. Is dat het geval dan vermeldt de printerhandleiding hoe het definiëren van een teken in z'n werk gaat. De in dit boek afgedrukte pixelbeelden bestaan uit van die zelf-gedefinieerde tekens.
Een aardige variant, die bij de beeldbewerking met kleuren al aan de orde is geweest, mag hier niet achterwege blijven. Door het beeld niet af te drukken in grijstinten, maar in lijndikten, ontstaat het effect dat in de reclamewereld veel wordt toegepast. Als printtekens worden dan lijntjes gebruikt met oplopende dikte.

Het oplossend vermogen van screen 3 van de MSX-computer is prachtig om er beelden mee vast te leggen ter ondersteuning van activiteiten op het scherm. Gedrukt op papier met een gewone printer gaat er veel van de charme van het uit pixels opgebouwde beeld verloren. De hierboven behandelde printmethoden dienen dan ook slechts beschouwd te worden als archivarisch hulpmiddel.

Een heel ander verhaal wordt het als we voor het afdrukken gebruik maken van een plotter en de methode waarbij lijntjes met verschillende diktes worden gebruikt.

De kwaliteit van het beeld is zonder meer vergelijkbaar met de beeldscherm-uitvoering. Eigenlijk staat het op papier extra leuk.

Het volgende plotprogramma is ontworpen voor de zeer laag geprijsde MSX-printer/plotter Toshiba HX-P570, die in dezelfde uitvoering ook onder een andere merknaam in de winkel kan worden aangetroffen.

```
 10 REM plotterafdruk
 20 DIM C(63,47)
 30 SCREEN 0:COLOR 1,10,10:CLS:KEY OFF
 40 FILES
 50 INPUT "Naam";N$
 60 LPRINT "Afdruk van ";N$
 70 LPRINT CHR$(&H1B)+"#"
 80 OPEN N$+".PIX" FOR INPUT AS #1
 90 FOR Y=0 TO 47
100   FOR X=0 TO 63
110     INPUT #1,C(X,Y)
120     IF C(X,Y)=12 THEN LPRINT "J4,0,0,1,-4,0,0,
        1,4,0":LPRINT"R0,-2"
130     IF C(X,Y)=2 THEN LPRINT "J4,0,0,1,-4,0":
        LPRINT"R0,-1"
140     IF C(X,Y)=3 THEN LPRINT "J4,0"
150     IF C(X,Y)=1 THEN LPRINT "R4,0"
160     IF C(X,Y)=0 THEN LPRINT "R4,0"
170   NEXT X
180   LPRINT "F":LPRINT "R0,17"
190 NEXT Y
200 CLOSE
```

De lijntjes worden getekend in de regels 120 tot en met 140 en hebben een lengte van vier beeldpunten. Het dikste lijntje is drie beeldpunten dik, het middenmaatje is twee beeldpunten dik en het dunste lijntje één. De beeldpuntafstand op de plotter is 0,2 mm. Dat heeft tot gevolg dat de totale afbeelding ongeveer 5 × 3 cm wordt. Een mooie afmeting om uw briefpapier een zeer persoonlijk cachet te geven. Wilt u een grotere afbeelding? Dan kunt u de diktes en de lengtes van de lijnen in de regels 120 tot en met 140 aanpassen. Houd er wel rekening mee dat de plotter er lang over doet om een grote afbeelding op papier te tekenen.

29 Iets over het geheugen

Vóór het beeldbewerkingsprogramma wordt opgebouwd, moeten we even stilstaan bij de geheugenruimte die voor onze experimenten beschikbaar is. Om hiervan een goed beeld te krijgen, is het nodig om te weten hoeveel geheugenruimte een compleet beeld ongeveer vergt.
Het multicolor-scherm bestaat uit 48 regels. Op elke regel is plaats voor 64 beeldpunten. In totaal moet de computer 3072 beeldpunten op kunnen slaan.
Elk beeldpunt heeft twee coördinaten en een kleur. Doordat we in dit boek de drie getallen samenvatten in een tweedimensionaal-array-element C(X,Y), slaan we per beeldpunt maar één getal op.
Een array-element neemt zonder verdere maatregelen acht bytes van het geheugen in beslag. Een beeld wordt dus beschreven door 3072 getallen die elk acht bytes in beslag nemen. Van het geheugen zouden dan ook 3072 × 8=24576 bytes moeten worden gereserveerd voor de opslag van een beeld. We hebben dus 24576/1024= 24 Kb (kilo-byte) nodig voor de opslag van een beeld. Het vrij beschikbare geheugen voor een 64K-machine is ongeveer 28 Kb. We hebben dan voor een programma slechts 4 Kb over. Dat is wel erg weinig als we in dat programma alle beeldmanipulatie-mogelijkheden die we al kennen, willen inbouwen.
In MSX-BASIC kunnen we variabelen ook in een ander jasje steken:

Gehele getallen	:	2 bytes per getal
Getallen met enkele precisie	:	4 bytes per getal
Getallen met dubbele precisie	:	8 bytes per getal

We zullen aan de hand van een miniprogramma demonstreren hoeveel geheugen voor een beeld nodig is als we de door ons gebruikte array-elementen onderbrengen in de verschillende categorieën. Als we geen maatregelen nemen, slaat de MSX-computer alle variabelen op met dubbele precisie (14 cijfers per getal).
Zodra we met de DIM-opdracht ruimte voor de opslag van een beeld reserveren, kunnen we uitrekenen hoeveel geheugen er nog voor het programma overblijft. De computer kan dat zelf ook.
Na het aanzetten van de computer verschijnt op het scherm een mededeling over de grootte van het beschikbare geheugen:

```
MSX BASIC version 1.Ø
Copyright 1983 by Microsoft
28815 Bytes free
```

We typen nu een regel in, waarin we de omvang van de array C(X,Y) vastleg-

gen en bovendien laten we de voor het programma en andere gegevens beschikbare geheugenruimte afdrukken:

```
DIM C(64,48):PRINT FRE(1) [Return]
42Ø8
```

Ongeveer 4 Kb, dat is niet veel voor een beetje uitgebreid programma.
We kijken nu hoe het verloopt als we de array C(X,Y) met enkele precisie op laten slaan. Dat kunnen we aangeven door direct achter de variabelenaam een uitroepteken te plaatsen:

```
DIM C!(64,48):PRINT FRE(1) [Return]
16495
```

Er is nu maar liefst ruim 16 Kb beschikbaar, dat scheelt aanzienlijk.
Wat we eigenlijk willen opslaan zijn kleurnummers, C(X,Y) staat immers voor de kleur van het beeldpunt X,Y. Nummers zijn gehele getallen. Als we de array C(X,Y) laten opslaan als een reeks gehele getallen ziet de uitkomst er als volgt uit:

```
DIM C%(64,48):PRINT FRE(1)
22683
```

Dat begint erop te lijken. De opslag van een beeld vraagt nu nog slechts 4 Kb. We hebben ongeveer 24 Kb over voor onze werkzaamheden. Dat is meer dan voldoende voor een zeer uitgebreid programma.
Anders ligt het bij de opslag op 3,5 inch-schijf. Bij het gebruik van C(X,Y), dus met dubbele precisie, vraagt een compleet beeld 17 Kb aan opslagruimte. Als C%(X,Y) wordt gebruikt, slaan we de kleurinformatie als een rij gehele getallen op. Toch vraagt dit nog altijd 16 Kb. (Dat is precies even veel als we voor de opslag van het videogeheugen nodig hebben.) De consequentie hiervan is dat er maar een beperkt aantal beelden op een schijf passen. Proberen we er per ongeluk te veel op te zetten, dan verschijnt op het beeldscherm de foutmelding: 'Device I/O error'. Laat het schijfje dan even zitten en reageer met het intypen van de CLOSE-opdracht. Daardoor kunnen er met deze schijf geen vreemde dingen gebeuren. Controleer dus voordat u beelden gaat opslaan, of er op de schijf genoeg ruimte is.

30 Beeldbewerking à la carte

De vele mogelijkheden om opgenomen beelden te modificeren, zijn in de vorige hoofdstukken de revue gepasseerd. In dit hoofdstuk bouwen we een menugestuurd programma op, waarmee vrijwel alle denkbare functies kunnen worden verricht. We beginnen met een schone lei. In dit geval dus met een lege cassetteband of een lege 3,5 inch-schijf.

Om te voorkomen dat we het programma niet meer kunnen bedienen als we het enige tijd niet hebben gebruikt en we zijn de exacte werking vergeten, maken we er een menu-gestuurd programma van dat aan duidelijkheid niets te wensen overlaat.

```
HOOFDMENU

1 Opnemen
2 Weergeven
3 Veranderen

Kies nummer
```

Vanaf dit hoofdmenu kunnen we een optie kiezen. Bij elke optie hoort een submenu. De inhoud van die submenu's zou er uit kunnen zien zoals hieronder afgebeeld:

```
MENU 1

Opnemen op:

1 Cassette
2 Quickdisk
3 Diskdrive

Kies nummer
```

Achter elke keuzemogelijkheid schuilt een subroutine die zorg draagt voor de juiste verrichting van de opdracht.

Het is onnodig ingewikkeld om het programma geschikt te maken voor het gebruik van alle in aanmerking komende opslagmedia. Iemand met een 3,5 inch-schijfeenheid zal weinig behoefte hebben aan een opslagmogelijkheid op cassette, terwijl bezitters van een zgn. Quickdisk weinig opschieten met een opslagmethode op 3,5 inch-diskette.

Het vorige hoofdstuk heeft aangetoond dat we niet zuinig met het beschikbare

geheugen om hoeven te springen. Toch zullen we terwille van de overzichtelijkheid het programma in vier afzonderlijke delen opsplitsen. Dat werkt gemakkelijk als we later veranderingen aan willen brengen. Bovendien duurt het laden van een programma-onderdeel korter dan het laden van het hele programma.
Om de opbouw van het programma helder te houden, beperken we ons tot het gebruik van de 3,5 inch-diskette als opslagmedium. Gebruikers van cassettebandjes kunnen de programma-onderdelen aanpassen. Dit komt er in het algemeen op neer dat voor een bestandsnaam 'CAS:' moet worden geplaatst. Een programmaregel zoals:

```
1Ø OPEN "PLAATJE.PIX" FOR INPUT AS #1
```

wordt dan:

```
1Ø OPEN "CAS:PLAATJE.PIX" FOR INPUT AS #1
```

Lees eerst uw BASIC-handleiding goed door om na te gaan hoe bestanden op uw opslagmedium opgeslagen c.q. van uw opslagmedium gelezen dienen te worden.
Vanwege de bijzondere kwaliteit van plotteruitvoer zal de in te bouwen printroutine niet alleen geschikt worden gemaakt voor printers (ongeacht het type), maar ook voor de plotter.
Het hoofdprogramma krijgt de naam 'PIXEL'. Dit programma geeft de mogelijkheid te kiezen uit:

- Het opnemen van beelden met de optische lezer die op de printer is gemonteerd.
- Het weergeven van beelden. Zowel beelden die als verzameling beeldpunten, als beelden die als videogeheugen zijn opgeslagen, kunnen met deze optie worden opgeroepen.
- Het bewerken van beelden. Door dit programma-onderdeel wordt een ander programma geladen waarmee beeldbewerking kan worden uitgevoerd.
- Het printen van beelden op een printer of plotten op een plotter. Behalve een 'hard copy' op de printer kan met het printprogramma ook een beeld op schijf worden gezet (videogeheugen-dump). Zoals eerder opgemerkt, gaat het laden van een beeld voor gebruik in andere programma's sneller als het is opgeslagen als video-geheugeninhoud.

Zet verse koffie en vergaar de nodige moed voor de nu komende omvangrijke klus: het intypen van de programma's. Hier en daar kunnen groepjes regels worden gekopieerd. Let wel op de juiste regelnummering. Foutzoeken achteraf is vaak tijdrovender en lastiger dan simpel typewerk. Leg de ingetypte programma's op diskette vast onder de programmanaam zoals die in de eerste REM-regel van het desbetreffende programma staat.

```
10  REM ************ PIXEL ******************
20  MAXFILES=2:KEY OFF
30  DIM C%(63,47)
40  OPEN"GRP:" FOR OUTPUT AS #1
50  SCREEN 2:COLOR 1,12,12:CLS
60  LINE(30,20)-(230,180),4,BF
70  LINE(10,5)-(80,40),10,BF
80  PSET(5,10),12
90  PRINT#1,"    M"
100 PSET(5,20),12
110 PRINT#1,"    S"
120 PSET(5,30),12
130 PRINT#1,"  PIXELS"
140 PSET(90,60),4:COLOR 15
150 PRINT#1,"HOOFDMENU"
160 PSET(90,80),5
170 PRINT#1,"1 Opnemen"
180 PSET(90,90),4
190 PRINT#1,"2 Weergeven"
200 PSET(90,100),4
210 PRINT#1,"3 Bewerken"
220 PSET(90,110),4
230 PRINT#1,"4 Printen"
240 PSET(120,150),4
250 PRINT#1,"Kies nummer"
260 A$=INKEY$:IF A$="" THEN 260
270 A=VAL(A$)
280 IF A<1 OR A>4 THEN BEEP:GOTO 260
290 ON A GOTO 320,940,300,310

300 RUN "BEWERK"
310 RUN "UITVOER"
320 CLS
330 LINE(30,20)-(230,180),10,BF
340 LINE(10,5)-(80,40),9,BF
350 COLOR 1
360 PSET(5,10),12
370 PRINT#1,"    M"
380 PSET(5,20),12
390 PRINT#1,"    S"
400 PSET(5,30),12
410 PRINT#1,"  PIXELS"
420 PSET(100,60),4
430 PRINT#1,"OPNEMEN"
440 PSET(40,80),10
450 PRINT#1,"* Afbeelding in printer"
460 PSET(80,135),10
470 PRINT#1," KIES KLEUR"
480 PSET(40,150),10
490 PRINT#1,"  1=Groen     2=Rood"
500 A$=INKEY$:IF A$="" THEN 500
510 A=VAL(A$)
520 IF A<1 OR A>2 THEN BEEP:GOTO 500
```

```
530 IF A=1 THEN C1%=12:C2%=2:C3%=3
    ELSE C1%=6:C2%=8:C3%=9
540 LINE(40,60)-(230,180),10,BF
550 PSET(40,120),10
560 PRINT #1,"* Printer op zelftest"
570 PSET(40,150),10
580 PRINT#1,"* Druk op spatiebalk"
590 IF NOT STRIG(0) THEN 590
600 SCREEN 3:COLOR 1,1,1:CLS
610 FOR Y=0 TO 47
620   IF NOT STRIG(1) THEN 620
630   FOR Q=1 TO 100:NEXT Q
640   IF STRIG(1) THEN 640
650   FOR X=12 TO 50
660     K=STICK(1)
670     C%(X,Y)=C1%
680     IF K=3 THEN C%(X,Y)=C1%
690     IF K=5 THEN C%(X,Y)=C2%
700     IF K=7 THEN C%(X,Y)=C3%
710     PSET(4*X,4*Y),C%(X,Y)
720     FOR Q=1 TO 2:NEXT Q:REM aan te passen
                                vertraging
730   NEXT X
740 NEXT Y
750 BEEP
760 COLOR 15
770 PSET(20,20),C1:PRINT#1,"opslaan"
780 PSET(0,100),C1:PRINT#1,"  (J/N)"
790 A$=INKEY$
800 IF A$="J" OR A$="j" THEN 820
810 IF A$="N" OR A$="n" THEN 50 ELSE 790
820 REM ********* PIXEL OPSLAG ****************
830 SCREEN 2:COLOR 1,10,10:CLS
840 GOSUB 1410
850 OPEN N$+".PIX" FOR OUTPUT AS #2
860 PRINT#2,C0%,C1%,C2%,C3%
870 FOR Y=0 TO 47
880   FOR X=0 TO 63
890     PRINT#2,C%(X,Y)
900   NEXT X
910 NEXT Y
920 CLOSE #2
930 SCREEN 0:COLOR 1,10,10:CLS
940 PRINT:"Plaats programmaschijf"
950 PRINT:"en druk op de spatiebalk"
960 IF NOT STRIG(0) THEN 960
970 GOTO 50

980  REM *********** WEERGEVEN ***************
990  SCREEN 2:COLOR 1,12,12:CLS
1000 LINE(30,20)-(230,180),9,BF
1010 LINE(10,5)-(80,40),13,BF
1020 PSET(5,10),12
1030 PRINT#1,"     M"
```

```
1040 PSET(5,20),12
1050 PRINT#1,"    S"
1060 PSET(5,30),12
1070 PRINT#1,"  PIXELS"
1080 PSET(90,60),9:COLOR 1
1090 PRINT#1,"WEERGEVEN van BEELDEN"
1100 PSET(40,95),9
1110 PRINT#1,"1 PIXELS  (.PIX)"
1120 PSET(40,115),9
1130 PRINT#1,"2 SCHERM  (.SCR)"
1140 PSET(120,160),9
1150 PRINT#1,"Kies nummer"
1160 A$=INKEY$:IF A$="" THEN 1160
1170 A=VAL(A$)
1180 IF A<1 OR A>2 THEN BEEP:GOTO 1160
1190 IF A=1 THEN T$=".PIX":GOSUB 1410
1200 IF A=1 THEN GOTO 1280
1210 IF A=2 THEN T$=".SCR":GOSUB 1410
1220 SCREEN 3:COLOR 1,1,1:CLS
1230 BLOAD N$+".SCR",S
1240 IF NOT STRIG(0) THEN 1240
1250 SCREEN 2:COLOR 1,12,12:CLS
1260 GOTO 50
1270 REM ******** BEELD OPROEPEN **********
1280 SCREEN 3:COLOR 1,1,1:CLS
1290 OPEN N$+".PIX" FOR INPUT AS #2
1300 INPUT #2,C0%,C1%,C2%,C3%
1310 FOR Y=0 TO 47
1320   FOR X=0 TO 63
1330     INPUT#2,C%(X,Y)
1340     PSET(4*X,4*Y),C%(X,Y)
1350   NEXT X
1360 NEXT Y
1370 CLOSE #2
1380 IF NOT STRIG(0) THEN 1380
1390 SCREEN 2:COLOR 1,12,12:CLS
1400 GOTO 50

1410 REM ************** SCHIJFINHOUD ************
1420 SCREEN 0:COLOR 1,10,10:CLS
1430 PRINT"    Plaats opslagschijf"
1440 PRINT:PRINT "     en druk op de spatiebalk"
1450 A$=INPUT$(1)
1460 CLS
1470 FILES
1480 INPUT "NAAM (zonder toevoeging)";N$
1490 CLS
1500 PRINT"Druk na beeldvorming op spatiebalk"
1510 FOR Q=1 TO 1000:NEXT Q
1520 SCREEN 2:COLOR 1,1,1:CLS
1530 RETURN
```

Dat was het hoofdprogramma. Door de verdeling in blokken is de loop van het programma goed te volgen. Als er vreemde dingen gebeuren, is er mogelijk ergens een %-teken weggevallen. De computer ziet C en C% als twee verschillende variabelen.
Het hoofdmenu wordt op het scherm gezet door het blok tussen de regels 10 en 290. Hier is dankbaar gebruik gemaakt van de mogelijkheid om ingekleurde blokken op het scherm te zetten met de LINE-opdracht. De schermopmaak bij een menu-gestuurd programma is belangrijk en door het gebruik van ondersteunende kleuren wordt een aantrekkelijk beeld gevormd.
In de regels 300 en 310 treffen we verwijzingen aan naar onderdelen die in een apart programma zijn gezet. Ze worden door het hoofdprogramma aangeroepen, zodat de indeling van het totale beeldbewerkingsprogramma – in afzonderlijke programma's – niet storend is. Ten gerieve van de cassetterecordergebruik(st)ers zijn de programma's die de beeldbewerking verzorgen op zichzelf staand uitgevoerd. Als van die programma's de plaats op de cassetteband bekend is, kunnen ze dus afzonderlijk worden ingeladen en gebruikt.
In de regels 320 tot en met 530 worden de voorbereidingen getroffen voor het opnemen van een beeld met de omgebouwde printer. De opname zelf wordt in het volgende blok gerealiseerd.
De regels 540 tot en met 810 vormen het software-deel van ons elektronicaproject (de beeldopnemer). Dit programmadeel houdt de stand van de eindschakelaar op de printer in de gaten en verwerkt de informatie van de sensor tot een beeld. Het is prachtig om het samenspel van dit programma-onderdeel en de printer te aanschouwen. Lijn voor lijn wordt de informatie vanaf het papier op het beeldscherm van de computer overgenomen.
Het nu verkregen beeld wordt in het massageheugen opgeslagen in de regels 820 tot en 930.
Het menu waaruit het beeldtype – dat we op het scherm willen zien – kan worden gekozen, is vastgelegd in de regels 940 tot en met 1230. Het oproepen van een beeld dat in beeldpunten is opgeslagen, duurt aanmerkelijk langer dan het oproepen van een beeld dat als inhoud van het videogeheugen is opgeslagen. Bovendien verschijnt het beeld bij gebruikmaking van de laatste methode in één keer op het scherm. Dit is zeer geschikt voor toepassingen in andere programma's (spelletjes, adventures enz.). In het nog volgende afdrukprogramma is de mogelijkheid ingebouwd om een beeld als videogeheugeninhoud op schijf op te slaan.
Een subroutine sluit het programma af. Het gedeelte tussen de regels 1240 en 1370 verzorgt het oproepen van het beeld. De schijf-inhoud wordt weergegeven met behulp van de subroutine die op regel 1380 begint.
Het nu volgende programma zorgt voor alle bewerkingen waarbij met kleuren en beeldpunten wordt gemanipuleerd. Het gedeelte waarmee plaatsveranderingen kunnen worden uitgevoerd, is als een afzonderlijk programma opgenomen.

```
10    REM ************ BEWERK ***************
20    MAXFILES=2
30    DIM C%(63,47)
40    OPEN"GRP:" FOR OUTPUT AS #1
50    SCREEN 2:COLOR 1,10,10:CLS
60    PSET(20,0),4
70    PRINT#1,"GEBRUIK DE CURSORBESTURING"
80    LINE(10,20)-(115,60),4,BF
90    LINE(130,20)-(235,60),4,BF
100   COLOR 15
110   PSET(40,22),4:?#1,"KLEUR"
120   PSET(20,35),4:?#1,"VERANDERING"
130   PSET(160,22),4:?#1,"PLAATS"
140   PSET(140,35),4:?#1,"VERANDERING"
150   Y=30:S=1
160   A=STICK(0)
170   IF A=7 THEN S=1
180   IF A=3 THEN S=2
190   COLOR 10:PSET(120,Y),10:PRINT#1,"■"
200   COLOR 1:PSET(120,Y),10:IF S=1 THEN PRINT#1,"<"
                                  ELSE PRINT#1,">"
210   IF STRIG(0) THEN 230
220   GOTO 160

230   IF S=2 THEN PSET(136,130),10:PRINT#1,
                      "WACHT !!!":RUN "GEO"
240   CLS:Y=130
250   LINE(45,120)-(200,160),4,BF
260   COLOR 15
270   PSET(50,130),4:PRINT#1,"Gehele beeld"
280   PSET(50,140),4:PRINT#1,"Individuele pixels"
290   A=STICK(0)
300   IF S=1 AND A=5 THEN Y=140
310   IF S=1 AND A=1 THEN Y=130
320   COLOR 10:PSET(30,Y),10:PRINT#1,"■"
330   COLOR 1:PSET(30,Y),10:PRINT#1,"*"
340   IF NOT STRIG(0) THEN 290
350   IF S=1 AND Y=130 THEN GOTO 370
360   IF S=1 AND Y=140 THEN GOTO 1320

370   REM ************ KLEURVERANDERING ***********
380   CLS:PRINT"           KLEUREN"
390   LINE(20,0)-(180,170),4,BF
400   COLOR 1:PSET(50,10),4:PRINT#1,">"
410   COLOR 15
420   PSET(30,10),4:PRINT#1," 0   Transparant"
430   PSET(30,20),4:PRINT#1," 1   Zwart"
440   PSET(30,30),4:PRINT#1," 2   Groen"
450   PSET(30,40),4:PRINT#1," 3   Lichtgroen"
460   PSET(30,50),4:PRINT#1," 4   Blauw"
470   PSET(30,60),4:PRINT#1," 5   Lichtblauw"
480   PSET(30,70),4:PRINT#1," 6   Donkerrood"
490   PSET(30,80),4:PRINT#1," 7   Ciaan"
500   PSET(30,90),4:PRINT#1," 8   Rood"
```

```
510  PSET(30,100),4:PRINT#1," 9   Lichtrood"
520  PSET(30,110),4:PRINT#1,"10   Geel"
530  PSET(30,120),4:PRINT#1,"11   Lichtgeel"
540  PSET(30,130),4:PRINT#1,"12   Donkergroen"
550  PSET(30,140),4:PRINT#1,"13   Paars"
560  PSET(30,150),4:PRINT#1,"14   Wit"
570  PSET(30,160),4:PRINT#1,"15   Helder wit"
580  C=0
590  FOR K=0 TO 3
600    PSET(10,180),10:COLOR 10
605    PRINT#1,"█████████████████████████████"
610    PSET(10,180),10:COLOR 1
615    PRINT#1,"Kies kleur ";K;"  (0=achtergrond)"
620    A=STICK(0)
630    FOR Q=1 TO 20:NEXT Q
640    C=C-(A=5)+(A=1)
650    IF C<0 THEN C=0
660    IF C>15 THEN C=15
670    COLOR 4:PSET(50,C*10+10),4:PRINT#1,"█"
680    COLOR 1:PSET(50,C*10+10),4:PRINT#1,">"
690    IF STRIG(0) THEN 700 ELSE 620
700    F(K)=C
710  NEXT K
720  C0%=F(0):C1%=F(1):C2%=F(2):C3%=F(3)
730  A=1
740  GOSUB 1020
750  GOSUB 1140
760  FOR Y=0 TO 47
770    FOR X=0 TO 63
780      IF C%(X,Y)=0 THEN C%(X,Y)=F(0)
790      IF C%(X,Y)=C1 THEN C%(X,Y)=F(1)
800      IF C%(X,Y)=C2 THEN C%(X,Y)=F(2)
810      IF C%(X,Y)=C3 THEN C%(X,Y)=F(3)
820      PSET(4*X,4*Y),C%(X,Y)
830    NEXT X
840  NEXT Y
850  IF NOT STRIG(0) THEN 850

860 SCREEN 0:COLOR 1,10,10:CLS
870 PRINT:PRINT:PRINT"   OPSLAAN (J/N) ";
880 A$=INPUT$(1)
890 IF A$="N" OR A$="n" THEN GOTO 1620
900 IF A$="J" OR A$="j" THEN GOSUB 1020 ELSE 880

910  REM ********* OPSLAG OP SCHIJF *************
920  CLS:PRINT"OPSLAAN van ";N$;".PIX"
930  OPEN N$+".PIX" FOR OUTPUT AS #2
940  PRINT#2,C0%,C1%,C2%,C3%
950  FOR Y=0 TO 47
960    FOR X=0 TO 63
970      PRINT#2,C%(X,Y)
980    NEXT X
990  NEXT Y
1000 CLOSE #2
1010 GOTO 1620
```

```
1020 REM ************* SCHIJFINHOUD *************
1030 SCREEN 0:COLOR 1,10,10:CLS
1040 PRINT"    Plaats opslagschijf"
1050 PRINT"    en druk op de spatiebalk"
1060 FOR Q=1 TO 50:NEXT Q
1070 IF NOT STRIG(0) THEN 1070
1080 CLS
1090 FILES:PRINT
1100 PRINT"Alleen sterren invullen"
1110 PRINT"Naam  ********.PIX"
1120 INPUT"      ";N$
1130 RETURN

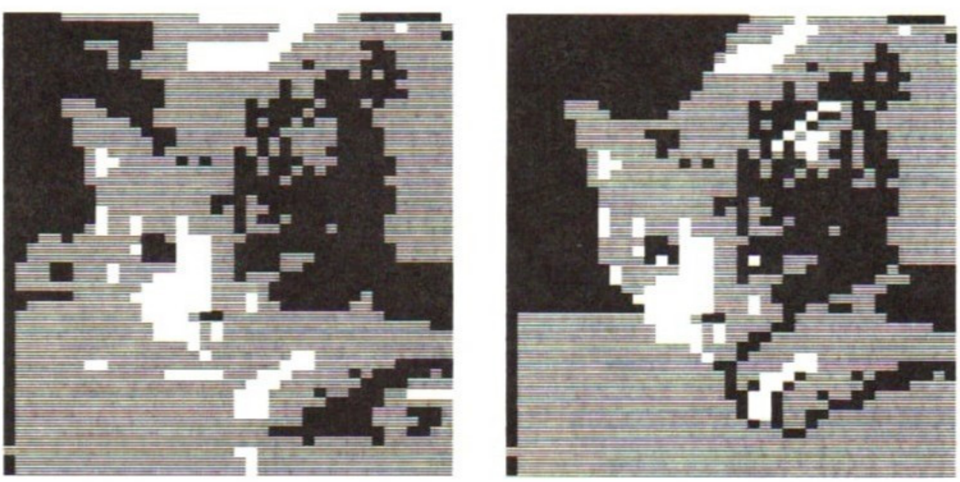
1140 REM ********** BEELD OPROEPEN **************
1150 CLS
1160 PRINT"DRUK NA BEELDVORMING OP SPATIEBALK"
1170 FOR Q=1 TO 1000:NEXT Q
1180 SCREEN 3:COLOR 1,1,1
1190 CLS
1200 OPEN N$+".PIX" FOR INPUT AS #2
1210 INPUT#2,C0%,C1%,C2%,C3%
1220 FOR Y=0 TO 47
1230   FOR X=0 TO 63
1240     INPUT#1,C%(X,Y)
1250     PSET(4*X,4*Y),C%(X,Y)
1260   NEXT X
1270 NEXT Y
1280 PRINT
1290 CLOSE #2
1300 IF NOT STRIG(0) THEN 1300
1310 RETURN

1320 REM *********** PIXEL VERANDERING ***********
1330 GOSUB 1020
1340 GOSUB 1140
1350 XO=0:YO=0:Y=0
1360 A=STICK(0)
1370 X=XO-(A=2)-(A=3)-(A=4)+(A=6)+(A=7)+(A=8)
1380 IF X<0 THEN X=0
1390 IF X>63 THEN X=63
1400 Y=YO+(A=1)+(A=2)+(A=8)-(A=4)-(A=5)-(A=6)
1410 IF Y<0 THEN Y=0
1420 IF Y>47 THEN Y=47
1430 PSET(4*X,4*Y),15
1440 FOR Q=1 TO 10:NEXT Q
1450 PSET(4*X,4*Y),1
1460 FOR Q=1 TO 10:NEXT Q
1470 IF STRIG(0) THEN GOSUB 1480
1480 IF X=XO AND Y=YO THEN 1360
1490 PSET(4*XO,4*YO),C%(XO,YO)
1500 XO=X:YO=Y
1510 GOTO 1360
```

```
1520 REM *********** KLEURVERANDERING ************
1530 A=STICK(0)
1540 IF A=1 THEN GOSUB 1020
1550 IF A=1 THEN GOTO 910
1560 T=T+(A=7)-(A=3)
1570 IF T<1 THEN T=1
1580 IF T>15 THEN T=15
1590 C%(X,Y)=T
1600 PSET(4*X,4*Y),C%(X,Y)
1610 IF STRIG(0) THEN RETURN ELSE 1530

1620 REM ********* TERUG NAAR HOOFDPROGRAMMA **********
1630 CLS
1640 PRINT "Plaats programmaschijf"
1650 PRINT "en druk op de spatiebalk"
1660 IF NOT STRIG(0) THEN 1620
1670 RUN "PIXEL"
```

Zoals we intussen gewend zijn, is ook hier weer een blokstructuur te herkennen. Het programma begint met een menu waaruit kan worden gekozen voor plaats- of kleurveranderingen (regels 10-200). Regel 230 verwijst naar een programma-onderdeel dat nog volgt.
Als we voor een kleurverandering kiezen, komen we in een submenu terecht waaruit kan worden gekozen voor de bewerking van het gehele beeld, of voor de bewerking van afzonderlijke beeldpunten. De selecties uit de menu's worden met behulp van de cursorbesturingstoetsen gemaakt.
De subroutines voor het weergeven van de schijfinhoud en het opslaan van de beelden kennen we al uit het vorige programma.
Op dit punt aangekomen, kunnen we beelden opnemen en weergeven. Bovendien kunnen de beelden worden gemanipuleerd. Wellicht is het een welkome onderbreking van het typewerk om eens een plaatje in het geheugen op te



Afbeelding 30-1 Een kat (links) en een geretoucheerde kat (rechts).

slaan en er vervolgens mee aan de slag te gaan. Vooral het bewerken van afzonderlijke pixels (beeldpunten) biedt interessante mogelijkheden. De plaatjes in afbeelding 30-1 laten hier iets van zien.
Het eerste beeld is weergegeven, precies zoals het is opgenomen. Het is opmerkelijk hoe weinig informatie de 'menselijke computer' (ook wel hersenen genoemd) nodig heeft om details te herkennen. Bekijk het prentje maar eens door uw oogharen vanaf enige afstand. De beeldpunten vloeien in elkaar over en het oorspronkelijke beeld wordt weer herkenbaar. Het tweede beeld is een bewerking van het eerste. Zo zijn in de ogen pretlichtjes aangebracht en zijn de linkervoorpoot en het linkeroor (voor de kijkers rechts) wat meer geaccentueerd.
De afbeelding wordt tijdens de opname min of meer centraal op het beeldscherm gezet. Die plaats is afhankelijk van de snelheid van de printer en de snelheid waarmee de beeldpunten op het beeldscherm van de computer worden gezet.
Het volgende programma maakt het mogelijk het beeld over het beeldscherm te schuiven, het beeld te spiegelen, te verdubbelen enz. Dit nieuwe beeld kan worden opgeslagen op schijf of cassette. De programmanaam 'GEO' slaat op de geometrische bewerkingen die het programma kan uitvoeren.

```
10    REM ******GEO*******************
20    KEY OFF:MAXFILES=2
30    DIM C%(63,47)
40    OPEN"GRP:" FOR OUTPUT AS #1
50    SCREEN 2:COLOR 1,12,12:CLS
60    LINE(130,20)-(235,192),4,BF
70    COLOR 15
80    PSET(160,22),4:PRINT#1,"Plaats"
90    PSET(140,35),4:PRINT#1,"verandering"
100   PSET(136,90),4:PRINT#1,"Hor/Vert"
110   PSET(136,100),4:PRINT#1,"Verdubbelen"
120   PSET(136,100),4:PRINT#1,"Omkeren"
130   PSET(136,120),4:PRINT#1,"Spiegelen"
140   PSET(136,130),4:PRINT#1,"Waterspiegel"
150   PSET(136,140),4:PRINT#1,"Speelkaart"
160   Y=90
170   A=STICK(0)
180   IF A=5 THEN Y=Y+10
190   IF Y>140 THEN Y=140
200   IF A=1 THEN Y=Y-10
210   IF Y<90 THEN Y=90
220   PSET(120,Y),12:COLOR 12:PRINT#1,"█"
230   PSET(120,Y),12:COLOR 1:PRINT#1,"*"
240   IF STRIG(0) THEN 260
250   GOTO 170
260   IF Y=90 THEN N=1
270   IF Y=100 THEN N=2
280   IF Y=110 THEN N=3
290   IF Y=120 THEN N=4
300   IF Y=130 THEN N=5
```

```
310  IF Y=140 THEN N=6
320  SCREEN 0:COLOR 1,10,10:CLS
330  ON N GOTO 340,650,790,910,1040,1160


340  PRINT"Verschuiving (links= -)"
350  PRINT"              (rechts=+)"
360  INPUT"Aantal beeldpunten ";N
370  CLS
380  PRINT"Verschuiving (Omhoog=-)"
390  PRINT"              (Omlaag=+)"
400  INPUT"Aantal beeldpunten ";M
410  GOSUB 1280
420  GOSUB 1540
430  IF N<0 THEN A=0:B=63:ST=1 ELSE A=63:B=0:ST=-1
440  IF M<0 THEN D=0:E=47:EP=1 ELSE D=47:E=0:EP=-1
450  FOR Y=D TO E STEP EP
460    FOR X=A TO B STEP ST
470      XN=X+N:YN=Y+M
480      PSET(4*XN,4*YN),C%(X,Y)
490      IF XN<0 THEN XN=0
500      IF XN>63 THEN XN=63
510      IF YN<0 THEN YN=0
520      IF YN>47 THEN YN=47
530      C%(XN,YN)=C%(X,Y)
540   NEXT X
550  NEXT Y
560  IF M<0 THEN D=48+M:E=47 ELSE D=0:E=M-1
570  FOR Y=D TO E
580    FOR X=0 TO 63
590      PSET(4*X,4*Y),1
600    NEXT X
610  NEXT Y
620  IF NOT STRIG(0) THEN 620
630  GOSUB 1280
640  GOTO 1390

650  REM ************ VERDUBBELEN ************
660  GOSUB 1280
670  GOSUB 1540
680  FOR Y=0 TO 47
690    FOR X=0 TO 32
700      PSET(4*X,4*Y),C%(X,Y)
710      PSET(4*(X+31),4*Y),C%(X,Y)
720      C%(X+31,Y)=C%(X,Y)
730    NEXT X
740  NEXT Y
750  IF NOT STRIG(0) THEN 750
760  GOSUB 1280
770  GOTO 1390

790  REM ********** OMKEREN *****************
800  GOSUB 1280
810  GOSUB 1540
```

```
820  FOR Y=0 TO 47
830    FOR X=0 TO 63
840      PSET(4*(63-X),4*Y),C%(X,Y)
850      C%(63-X,Y)=C%(X,Y)
860    NEXT X
870  NEXT Y
880  IF NOT STRIG(0) THEN 880
890  GOSUB 1280
900  GOTO 1390

910  REM *********** SPIEGELEN *************
920  GOSUB 1280
930  GOSUB 1490
940  FOR Y=0 TO 47
950    FOR X=0 TO 32
960      PSET(4*X,4*Y),C%(X,Y)
970      PSET(4*(62-X),4*Y),C%(X,Y)
980      C%(62-X,Y)=C%(X,Y)
990    NEXT X
1000 NEXT Y
1010 IF NOT STRIG(0) THEN 1010
1020 GOSUB 1280
1030 GOTO 1390

1040 REM *************WATERSPIEGEL ***********
1050 GOSUB 1280
1060 GOSUB 1540
1070 FOR Y=0 TO 23
1080   FOR X=0 TO 63
1090     PSET(4*X,4*(47-Y)),C%(X,Y)
1100     C%(X,47-Y)=C%(X,Y)
1110   NEXT X
1120 NEXT Y
1030 IF NOT STRIG(0) THEN 1030
1140 GOSUB 1280
1150 GOTO 1390


1160 REM ************* SPEELKAART *************
1170 GOSUB 1280
1180 GOSUB 1540
1190 FOR Y=0 TO 23
1200   FOR X=0 TO 63
1210     PSET(4*(63-X),4*(47-Y)),C%(X,Y)
1220     C%(63-X,47-Y)=C%(X,Y)
1230   NEXT X
1240 NEXT Y
1250 IF NOT STRIG(0) THEN 1250
1260 GOSUB 1280
1270 GOTO 1390


1280 REM ******* BESTANDEN LEZEN ************
1290 SCREEN 0:COLOR 1,10,10:CLS
```

```
1300 PRINT "Plaats opslagschijf"
1310 PRINT "en druk op de spatiebalk"
1320 IF NOT STRIG(0) THEN 1320
1330 CLS
1340 FILES
1350 PRINT"Alleen sterren invullen"
1360 PRINT"Naam  ********.PIX"
1370 INPUT"       ";N$
1380 RETURN

1390 REM ********** BEELDOPSLAG *****************
1400 CLS:PRINT "OPSLAAN ";N$;".PIX"
1410 OPEN N$+".PIX" FOR OUTPUT AS #2
1420 PRINT#2,C0%,C1%,C2%,C3%
1430 FOR Y=0 TO 47
1440   FOR X=0 TO 63
1450     PRINT#2,C%(X,Y)
1460   NEXT X
1470 NEXT Y
1480 CLOSE #2
1490 SCREEN 0:COLOR 1,10,10:CLS
1500 PRINT "Plaats programmaschijf"
1510 PRINT "en druk op spatiebalk"
1520 IF NOT STRIG(0) THEN 1520
1530 RUN "PIXEL"

1540 REM ******** BEELD INLEZEN *****************
1550 CLS:PRINT"Druk NA BEELDVORMING op spatiebalk"
1560 FOR Q=1 TO 1000:NEXT Q
1570 SCREEN 3:COLOR 1,1,1:CLS
1580 OPEN N$+".PIX" FOR INPUT AS #2
1590 INPUT#2,C0%,C1%,C2%,C3%
1600 FOR Y=0 TO 47
1610   FOR X=0 TO 63
1620     INPUT#2,C%(X,Y)
1630     PSET(4*X,4*Y),C%(X,Y)
1640   NEXT X
1650 NEXT Y
1660 CLOSE #2
1670 IF NOT STRIG(0) THEN 1670
1680 RETURN
```

Ongetwijfeld herkent u de reeds behandelde beeldbewerkingen, zoals spiegelen en omkeren. Ook wat het opslaan en lezen van de beelden betreft, is er weinig nieuws onder de zon.

De drie grootste programma's zijn nu klaar. Afbeeldingen kunnen in alle mogelijke kleuren en standen op het scherm worden gezet. Het enige dat nog ontbreekt is het maken van een 'hard copy'. Zoals afgesproken, maken we het afdrukprogramma geschikt voor:

- het maken van afdrukken op een printer,
- het maken van afdrukken op een plotter en
- het 'dumpen' van de video-geheugeninhoud op diskette.

```
10   REM ***********UITVOER**************
20   MAXFILES=2
30   DIM C%(63,47)
40   OPEN"GRP:" FOR OUTPUT AS #1
50   SCREEN 2:COLOR 1,12,12:CLS:KEY OFF
60   LINE(30,20)-(230,180),4,BF
70   LINE(10,5)-(80,40),10,BF
80   PSET(5,10),12
90   PRINT#1,"    M"
100  PSET(5,20),12
110  PRINT#1,"    S"
120  PSET(5,30),12
130  PRINT#1," PIXELS"
140  PSET(90,60),4:COLOR 15
150  PRINT#1,"UITVOER"
160  PSET(90,80),4:PRINT#1,"1 Dump op schijf"
170  PSET(90,95),4:PRINT#1,"2 Printer"
180  PSET(90,110),4:PRINT#1,"3 Plotter"
190  PSET(120,150),4:PRINT#1,"Kies nummer"
200  A$=INKEY$:IF A$="" THEN 200
210  A=VAL(A$)
220  IF A<1 OR A>3 THEN BEEP:GOTO 200
230  ON A GOTO 240,390,440

240  REM *****3,5 INCH-SCHIJF*******************
250  SCREEN 0:COLOR 1,10,10:CLS
260  PRINT "Plaats de opslagschijf voor"
280  PRINT "het beeld pas NADAT het beeld"
300  PRINT "volledig gevormd is"
310  PRINT
320  PRINT"DRUK DAARNA OP DE SPATIEBALK"
330  FOR Q=1 TO 2000:NEXT Q
340  GOSUB 740
350  GOSUB 830
360  IF NOT STRIG(0) THEN 360
370  BSAVE N$+".SCR",0,16383,S
380  GOTO 1030

390  REM *********** PRINTER ***************
400  GOSUB 740
410  GOSUB 830
420  PR=1
430  GOTO 1030

440  REM ********** PLOTTER ***************
450  GOSUB 740
460  SCREEN 0:COLOR 1,10,10:CLS
470  PRINT"Beeldhoogte 3 of 6 centimeter";
480  A$=INPUT$(1)
490  IF A=3 THEN A=1:GOTO 520
500  IF A=6 THEN A=2:GOTO 520
510  BEEP:GOTO 460
520  CLS:PRINT"PLOTTER UITVOER"
530  OPEN N$+".PIX" FOR INPUT AS #2
```

```
540  INPUT#2,C0%,C1%,C2%,C3%
550  LPRINT N$
560  LPRINT CHR$(&H1B)+"#"
570  FOR Y=0 TO 47
580    FOR R=1 TO A
590      FOR X=0 TO 63
600        IF R=2 THEN 620
610        INPUT#2,C%(X,Y)
620        FOR P=1 TO A
630          IF C%(X,Y)=C1% THEN LPRINT "J4,0,0,1,-
                         4,0,0,1,4,0":LPRINT"R0,-2"
640          IF C%(X,Y)=C2% THEN LPRINT"J4,0,0,1,-
                         4,0":LPRINT"R4,-1"
650          IF C%(X,Y)=C3% THEN LPRINT"J4,0"
660          IF C%(X,Y)=C0% THEN LPRINT"R4,0"
670        NEXT P
680      NEXT X
690      LPRINT"F":LPRINT"R0,17"
700    NEXT R
710  NEXT Y
720  CLOSE #2
730  GOTO 1030

740  REM ********* BESTANDEN LEZEN *************
750  SCREEN 0:COLOR 1,10,10:CLS
760  PRINT "Plaats opslagschijf"
770  PRINT "en druk op de spatiebalk"
780  IF NOT STRIG(0) THEN 780
790  CLS
800  FILES
810  INPUT "NAAM ";N$
820  RETURN

830  REM ************* BEELD UITLEZEN **********
840  SCREEN 3:COLOR 1,1,1:CLS
850  OPEN N$+".PIX" FOR INPUT AS #2
860  INPUT#2,C0%,C1%,C2%,C3%
870  FOR Y=0 TO 47
880    FOR X=0 TO 63
890      INPUT#2,C%(X,Y)
900      IF NOT PR THEN 950
910      IF C%(X,Y)=C0% THEN P$=" "
920      IF C%(X,Y)=C1% THEN P$="#"
930      IF C%(X,Y)=C2% THEN P$="="
940      IF C%(X,Y)=C3% THEN P$="-"
950      LPRINT P$;
960      PSET(4*X,4*Y),C%(X,Y)
970    NEXT X
980    IF NOT PR THEN 1000
990    LPRINT
1000 NEXT Y
1010 CLOSE #2
1020 RETURN
1030 SCREEN 0:COLOR 1,10,10:CLS
```

```
1040 PRINT:"Plaats programmaschijf"
1050 PRINT "en druk op spatiebalk"
1060 IF NOT STRIG(0) THEN 1060
1070 RUN "PIXEL"
```

Het is u misschien opgevallen dat er in het programmablok PRINTER niets wordt geprint. Dat gebeurt in werkelijkheid tegelijk met het inlezen van het beeld. Het lijkt inconsequent, maar deze opzet bespaart wat werk. Als u veranderingen in de programma's wilt aanbrengen, bent u daar natuurlijk vrij in. Omdat het plotterdeel er wat eigenaardig uitziet, kunnen er gemakkelijk tikfoutjes insluipen. Afbeelding 30-2 laat zien hoe de plotter dunne en dikke lijntjes maakt.



Afbeelding 30-2 Plotpatronen.

Een plotterafbeelding is drie centimeter hoog. Door het fijne lijnenpatroon blijven de details goed zichtbaar. Misschien is zo'n plotterafbeelding nog wel mooier dan de afbeelding op het beeldscherm... Willen we een afbeelding tweemaal zo groot maken (zes centimeter hoog), dan laten we de plotterpatronen tweemaal naast elkaar en iedere regel tweemaal onder elkaar afdrukken. (Op deze manier worden de blokjes waaruit de afbeelding bestaat, vier keer zo groot.) Op A4-papier kan de afbeelding eventueel nog groter worden getekend. Geef de variabele A in regel 620 de waarde 3 (of 4). De tijd die nodig is om het beeld te plotten, wordt hierdoor wel aanzienlijk vergroot.
Voor een MSX-printer worden met de tekens

≡ [SHIFT][GRAPH]=
= is gelijk-teken
– minteken

de beste resultaten verkregen.
Voor andere printers zou u de tekens #, = en – kunnen gebruiken. Het beste kunt u wat experimenteren met verschillende tekens. Het mooiste resultaat verkrijgt u door zelf karakters te definiëren. Deze worden als bit-patronen naar de printer verzonden. Raadpleeg de handleiding van uw printer hierover. Een juiste instelling van de pitch en de regelafstand garanderen een (bijna) naadloze aaneensluiting van de tekens.
Een andere manier om een 'hard copy' van het beeldscherm te maken, is het fotograferen van het beeldscherm. De beeldschermen laten zich goed vastleggen op dia's of foto's. Door het enigszins gebolde beeldscherm kan er gemakkelijk randvertekening ontstaan, vooral als we het scherm vanaf een kleine afstand fotograferen. Door een lens met een wat langer brandpunt te nemen, wordt de afstand tot het scherm al gauw groter, waardoor de vertekening verdwijnt. Let er wel op dat het beeldscherm goed schoon is: op een foto vallen allerlei vlekjes, vuiltjes en vingerafdrukken extra op.

Het TV-beeld wordt 50 keer per seconde herhaald. Oudere fototoestellen hebben nog een sluitertijd van 1/25 seconde. In die tijd wordt het beeld precies tweemaal geschreven. De modernere camera's hebben een aan de internationale normen aangepaste sluitertijd van 1/30 seconde. Als met deze sluitertijd een opname wordt gemaakt, zien we een hinderlijke streep, omdat de beeldvorming nog niet compleet is. Beter is het dan ook om een lange sluitertijd te nemen. In dat geval nemen we tientallen keren hetzelfde beeld op en is het aandeel van een nog niet voltooid beeld te verwaarlozen. De sluitertijd die het beste voldoet, is een halve seconde. Dit houdt wel in dat het fototoestel op een statief moet staan. Het diafragma zal een instelling tussen 8 en 22 moeten hebben, afhankelijk van de omstandigheden. Probeer de opname beeldvullend te maken, zodat de schermranden niet zichtbaar zijn (die kunt u overigens ook later van de foto afknippen). Door de film tweemaal te belichten – eenmaal met een titel en eenmaal met een gedigitaliseerd beeld – kunnen prachtige aankondigingen in diaseries worden gemaakt. Ook voor zelfgemaakte videofilms zijn opgeslagen computerbeelden uitermate geschikt.

31 Beeldmanipulatie met de MSX2-computer

Als we denken aan grafische computertoepassingen, kunnen we niet om de MSX2-computer heen. In dit hoofdstuk zullen we even proeven aan de mogelijkheden die deze fantastische machine biedt. Het is niet moeilijk om de door ons opgenomen plaatjes op het hogeresolutiescherm van de MSX-computer af te beelden, maar wel op dat van de MSX2-computer. Het programma PIXEL is gemakkelijk aan te passen: vervang alle SCREEN 3-opdrachten in SCREEN 2-opdrachten. In de PSET-opdrachten moeten de factoren 4 worden verwijderd, anders komen de afgebeelde puntjes te ver uit elkaar te liggen. Dus niet PSET(4*X,4*Y),C%(X,Y) maar PSET(X,Y),C%(X,Y).
Het nu verkregen beeld is maar 63 bij 47 beeldpunten groot, pasfotoformaat. Het is echter wel van zeer goede kwaliteit. Zo'n pasfoto doet het goed in andere programma's. Met een MSX2-computer kunnen we zo'n scherp prentje op iedere plaats op het hogeresolutiescherm zetten door bij de coördinaten van de beeldpunten, de coördinaten van de gewenste linkerbovenhoek op te tellen. De opdracht PSET(X+100,Y+70),C%(X,Y) zet het plaatje ongeveer in het midden van het beeldscherm.
Het inlezen van een beeld gaat relatief langzaam. Het is met een MSX-computer niet zo aantrekkelijk om de beelden over het scherm te laten schuiven. Met de MSX2-computer daarentegen wel. Deze machine biedt unieke grafische mogelijkheden die in veel opzichten vergelijkbaar zijn met de fraaie staaltjes van grafische effecten die omroeporganisaties gebruiken in hun aankondigingen.
MSX2-BASIC heeft extra beeldbewerkingsopdrachten, zoals SET PAGE en COPY.
Het volgende programma is ontleend aan het eerder behandelde programma PIXEL. Het is echter sterk ingekort en niet in de tot nu toe opgebouwde programmacluster opgenomen. Zie het als een demonstratieprogramma voor de MSX2-computerbezit(s)ters. Het beeld wordt weergegeven op één van de hogeresolutieschermen. In het programma is SCREEN 5 gebruikt.

```
10  REM MSX2
20  MAXFILES=2:KEY OFF
30  DIM C%(63,47)
40  SCREEN5:SET PAGE 1,1:COLOR 1,10,10:CLS
50  OPEN"GRP:"FOR OUTPUT AS #1
60  T$="PIX":GOSUB 500
70  SCREEN 5:COLOR 1,1,1:CLS
80  SET PAGE 0,0
90  OPEN"A:"+N$+".PIX" FOR INPUT AS #2
100 INPUT #2,C0%,C1%,C2%,C3%
110 FOR Y=0 TO 47
120   FOR X=0 TO 63
```

```
130       INPUT#2,C%(X,Y)
140       PSET(X,Y),C%(X,Y)
150    NEXT X
160 NEXT Y
170 CLOSE #2
180 REM manipulatie
190 COPY (0,0)-(63,47),0 TO (0,0),1
200 FOR X=0 TO 3
210    FOR Y=0 TO 3
220       COPY (0,0)-(63,47) TO (X*63,Y*47)
230       FOR Q=1 TO 200:NEXT Q
240    NEXT Y
250 NEXT X
260 FOR Q=1 TO 2500:NEXTQ
270 SET PAGE 1,1
280 FOR I=1 TO 4
290    COPY (0,0)-(63,47),0 TO (40*I,34*I)
300    FOR Q=1 TO 100:NEXT Q
310 NEXT I
320 FOR Q=1 TO 2000:NEXT Q
330 FOR N=1 TO 5
340    SET PAGE 0,0
350    FOR Q=1 TO 500:NEXT Q
360    SET PAGE 1,1
370    FOR Q=1 TO 500:NEXT Q
380 NEXT N
390 FOR I=1 TO 200
400    COPY(0,0)-(63,47),0 TO (I,0)
410 NEXT I
420 FOR I=1 TO 200
430    COPY(0,48)-(63,96),0 TO (I,48)
440 NEXT I
450 FOR I=1 TO 200
460    COPY(0,97)-(63,212),0 TO (I,97)
470 NEXT I
480 SET PAGE 0,0
490 IF STRIG(0) THEN END ELSE GOTO 490
500 REM laat schijfinhoud zien
510 SCREEN0:COLOR 1,10,10:CLS
520 PRINT"Plaats opslagdiskette..."
530 PRINT:PRINT:PRINT"en druk op spatiebalk ";
540 A$=INPUT$(1)
550 CLS
560 FILES
570 PRINT"Alleen sterren invullen"
580 PRINT"Naam  ********.";T$
590 INPUT"     ";N$
600 CLS
610 SCREEN2:COLOR 1,1,1:CLS
620 RETURN
```

In de programmaregels 10 tot en met 170 wordt het beeld in de linkerbovenhoek van het beeldscherm geplaatst. Vanuit de subroutine aan het einde van het programma, maakt de gebruik(st)er een keuze uit de af te beelden plaatjes. Let even op regel 40. Hier wordt kennelijk nogal overbodig SCREEN 5 aangeroepen. Toch is deze regel belangrijk. In SCREEN-modus 5 kunnen we vier onafhankelijke schermbeelden opslaan en bewerken. Deze beelden staan op pagina (PAGE) 0 tot en met 3. Als de SCREEN 5-opdracht wordt uitgevoerd, wordt alleen de eerste pagina gewist. Alle andere pagina's kunnen dus nog beelden bevatten uit eerder uitgevoerde programma's. Omdat we in dit voorbeeld twee pagina's gebruiken, wordt in regel 40 pagina 1 voor alle zekerheid gewist. Elke nieuwe programma-run begint met een schone lei. Bovendien geven we de achtergrond alvast een kleurtje.

```
SCREEN 5:SET PAGE 1,1:COLOR 1,1Ø,1Ø:CLS
```

Het eerste cijfer in de SET PAGE-opdracht geeft aan welke pagina op het scherm wordt afgebeeld. Het tweede cijfer bepaalt welke pagina in bewerking is. Het is mogelijk om een plaatje af te beelden, terwijl intussen veranderingen aan een ander worden aangebracht. Dit plaatje staat in het videogeheugen. SET PAGE 0,0 wil zeggen dat beeldpagina 0 op het scherm staat en dat bovendien alle beeldmanipulaties op dit beeld betrekking hebben. SET PAGE 1,0 wil zeggen dat we pagina 1 zien en dat de beeldbewerkingen plaatsvinden op de onzichtbare pagina 0.
Behalve de SET PAGE-opdracht speelt ook de COPY-opdracht een belangrijke rol in dit programma. In regel 190 wordt een deel van beeldpagina 0 naar een positie op de nog onzichtbare pagina 1 gekopieerd. Als we achter de coördinaten van de COPY-opdracht aangeven om welke pagina het gaat, is het niet belangrijk welke pagina op dat ogenblik op het scherm staat. Als we naar het huidige beeldscherm willen kopiëren, hoeven we geen paginacijfer op te geven.
We zullen de loop van het programma eens nagaan. De regels 180 tot en met 250 bevatten twee lussen. Eerst wordt het beeld viermaal gekopieerd. Telkens verandert de Y-coördinaat: er komen vier beeldjes onder elkaar te staan. Vervolgens wordt de X-coördinaat verhoogd en worden opnieuw vier beeldjes onder elkaar afgebeeld. Dit gaat zo door totdat er vier rijtjes van vier beelden op het scherm staan. Omdat er geen paginanummer achter de COPY-opdracht staat, gebeurt dit alles op de zichtbare pagina (PAGE 0). Als we regel 230 weglaten, gebeurt het kopiëren zó snel dat het voor ons oog niet meer bij te houden is: het lijkt alsof alle zestien plaatjes in één keer op het scherm worden gezet. Door de vertraging in regel 230 ontstaat het effect alsof de prentjes als kaarten over een tafel worden uitgespreid.
Regel 270 maakt PAGE 1 zichtbaar. We hadden er al een beeldje ingezet toen de pagina nog niet zichtbaar was (regel 90). Nu plaatsen we er op de diagonaal van het beeldscherm nog wat afbeeldingen bij (regel 280-310).
Eenmaal opgeslagen beelden kunnen met een ongekende snelheid op het beeldscherm worden gezet. De programmaregels 330 tot en met 380 laten pa-

gina 0 en pagina 1 vijfmaal afwisselend op het scherm verschijnen. Hoe waanzinnig snel dit kan, kunt u zien als u de vertragingen uit de regels 350 en 370 verwijdert. Laat het beeld 100 keer wisselen.
Animatie met onze grafische juweeltjes behoort ook tot de MSX2-mogelijkheden. We kunnen de beeldjes als sprites over het scherm laten bewegen. Het laatste deel van het programma laat hier enkele voorbeelden van zien.
De snelheid waarmee de MSX2-computer met gecompliceerde beelden omgaat, is verbazingwekkend. Dat er nog veel meer mogelijkheden in deze machine schuilen, bewijzen twee opdrachten (die niet in alle handboeken zijn opgenomen). MSX2-BASIC kent de opdracht COPY SCREEN [<mode>] waarmee externe videobeelden gedigitaliseerd in het geheugen kunnen worden opgeslagen. De opdracht SET VIDEO <mode> [,<ym>[,<cb>[,<sync> [,<audio>[,<video input>[,<AV control>]]]]]] laat zien dat er rekening is gehouden met koppeling aan video-apparatuur. Met SET VIDEO kunnen verschillende beelden worden 'gemixed' (superimpose). De verschillende parameters maken een flexibele aanpassing mogelijk:

<mode>	0 tot 3	(waar komen de beelden vandaan)
<ym>	0 of 1	(beeldintensiteit half of vol)
<cb>	0 of 1	(kleurenbesturing voor invoer of uitvoer)
<sync>	0 of 1	(interne of externe synchronisatie)
<audio>	0 tot 3	(regeling audiosignaal)
<video>	0 of 1	(invoer via RGB- of TV-aansluiting)
<AV>	0 of 1	(audio/video-output via RGB-aansluiting)

Over het concrete gebruik van de computer/video-koppeling is nog weinig bekend. Het programma in dit hoofdstuk heeft echter laten zien dat er nog steeds een uitdagend terrein van grafische mogelijkheden voor ons open ligt.

32 Tot slot

In dit gedeelte van het boek heeft u kennis kunnen maken met de intrigerende wereld van beeldbewerking. We hebben hierbij gebruik gemaakt van het lageresolutiescherm. Voor gecompliceerde taken op het hogeresolutiescherm leent de MSX-computer zich minder goed; de MSX2-computer daarentegen wel. Echter, als u rekening houdt met de regel dat op het hogeresolutiescherm telkens acht beeldpunten naast elkaar dezelfde kleur moeten hebben, kunt u ook op een MSX-computer van het hogeresolutiescherm gebruikmaken en de programma's daaraan aanpassen.
Het oplossend vermogen van de in dit boek beschreven beeldopnemer kan drastisch worden verhoogd door gebruik te maken van lenzen. Als u die wilt toepassen, kunt u ook het beste gebruik maken van een gescheiden zend- en ontvangdiode. Als u geen financiële beperkingen kent, is de toepassing van een opnemer uit een streepjescodelezer ideaal. De printer moet dan wel in staat zijn om kleine regelverschuivingen uit te voeren.
Ondanks de mogelijke verbeteringen is er geen reden om ontevreden te zijn met het behaalde resultaat. Het brengt beeldbewerkingstechnieken binnen de mogelijkheden van de homecomputer-bezit(s)ter en geeft daarmee een nieuwe dimensie aan haar of zijn werkzaamheden met dit fascinerende apparaat.

Het maken van grafische afbeeldingen is één van de leukste dingen die men op een computer kan doen. De vele MSX-computerbezitters kunnen aan dit boek dan ook een hoop plezier beleven.
'Grafische experimenten voor MSX-computers' is uit twee delen opgebouwd.
Het eerste deel geeft een reeks korte BASIC-programma's die de grafische mogelijkheden van de MSX-computer op speelse wijze illustreren. Verder biedt het een eenvoudig tekenprogramma en een sprite-ontwerpprogramma. De auteur heeft de programma's aan elkaar geregen zodat de uitvoering ervan een imposante MSX-show teweegbrengt.
Het tweede deel beschrijft hoe we een beeldbewerkingssysteem kunnen bouwen. De hardware van dit systeem bestaat uit een beeld-digitizer, de software bestaat uit beeldverwerkingsprogramma's. Met behulp hiervan kunnen foto's en tekeningen – in digitale code – in de computer worden geladen. We krijgen de foto of tekening opgebouwd uit blokjes op het beeldscherm te zien. Deze 'pixelbeelden' kunnen we spiegelen, kleuren, bijwerken, verdubbelen enz. Zo ontstaan er prachtige plaatjes die we bijvoorbeeld als achtergrond in een computerspel kunnen gebruiken.

ISBN 90 201 1967 2